(12)

PROCEEDINGS:

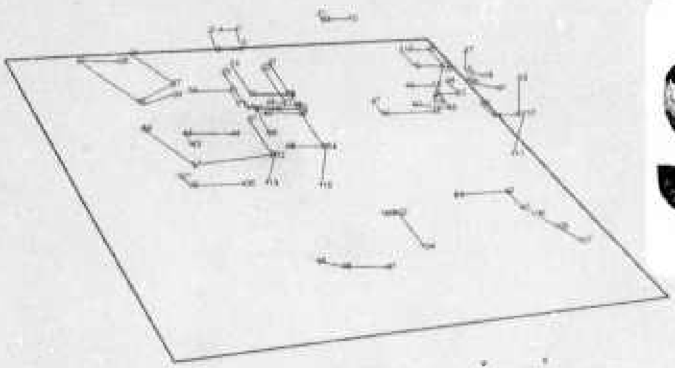
IMAGE UNDERSTANDING WORKSHOP

JUNE 1983

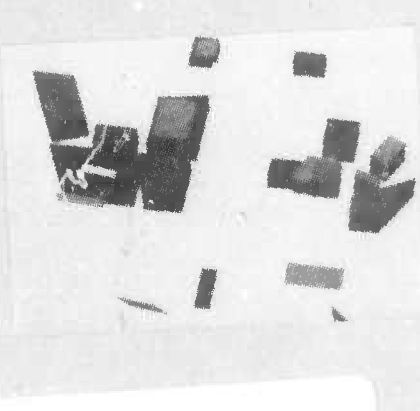
Sponsored by:
Information Processing Techniques Office
Defense Advanced Research Projects Agency

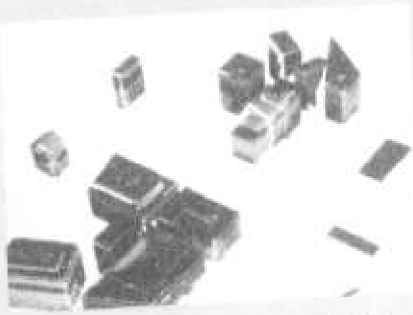
Conducted in Conjunction with:
Computer Vision and Pattern
Recognition Conference of the
IEEE Computer Society

Urban Imagery



Wire-Framed Description





Science Applications, Inc.

COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Image Understanding: Proceedings of a Workshop (14th) Held at
Arlington, Virginia on June 23, 1983.

(SOURCE): Science Applications, Inc., McLean, VA.

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A130 251.

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#:	TITLE:
AD-P001 189	MIT Progress in Understanding Images.
AD-P001 190	The SRI Image Understanding Research Program.
AD-P001 191	Intelligent Vision Systems.
AD-P001 192	Image Understanding Research at the University of Massachusetts.
AD-P001 193	Surface Constraints from Linear Extents.
AD-P001 194	Computing Visual Correspondence.
AD-P001 195	Viewframes: A Connectionist Model of Form Perception.
AD-P001 196	The Use of Difference Fields in Processing Sensor Motion.
AD-P001 197	The Facet Approach to Optic Flow.
AD-P001 198	Special Purpose Automatic Programming for 3D Model-Based Vision.
AD-P001 199	MAPS: The Organization of a Spatial Database System Using Imagery, Terrain, and Map Data.
AD-P001 200	Segment-Based Stereo Matching.
AD-P001 201	Software Metrics for Performance Analysis of Parallel Hardware.
AD-P001 202	On the Evaluation of Scene Analysis Algorithms.
AD-P001 203	Image Understanding Application Projects: Implementation Progress Report.
AD-P001 204	Robot Vehicles: A Survey and Proposed Test-Bed Facility.
AD-P001 205	Kinematics of Image Flows.
AD-P001 206	Fractal-Based Description of Natural Scenes.
AD-P001 207	Rule-Based Strategies for Image Interpretation.
AD-P001 208	The Perceptual Organization of Visual Images: Segmentation as a Basis for Recognition.
AD-P001 209	The Theory of Straight Homogeneous Generalized Cylinders.
AD-P001 210	An Algorithm to Display Generalized Cylinders.
AD-P001 211	Perceptual Organization and Curve Partitioning.

COMPONENT PART NOTICE (CON'T)

AD#:	TITLE:
AD-P001 212	Scene Matching by Hierarchical Correlation.
AD-P001 213	The Relationship Between Image Irradiance and Surface Orientation.
AD-P001 214	Environmental Relations in Image Understanding: The Force of Gravity.
AD-P001 215	Adapting Optical-Flow to Measure Object Motion in Reflectance and X-Ray Image Sequences.
AD-P001 216	Processing Restricted Sensor Motion.
AD-P001 217	Stereo Matching from the Topological Viewpoint.
AD-P001 218	Extraction of Textured Regions in Aerial Imagery.
AD-P001 219	The Topographic Primal Sketch and its Application to Passive Navigation.
AD-P001 220	Monocular Reconstruction of a Complex Urban Scene in the 3D MOSAIC System.
AD-P001 221	Progress in Stereo Mapping.
AD-P001 222	A Content Addressable Array Parallel Processor and Some Applications.

IMAGE UNDERSTANDING

Proceedings of a Workshop
Held at
Arlington, Virginia
June 23, 1983

Sponsored by the
Defense Advanced Research Projects Agency

Science Applications, Inc.
Report Number SAI-84-176-WA
Lee S. Baumann
Workshop Organizer and
Proceedings Editor

This report was supported by
The Defense Advanced Research
Projects Agency under DARPA
Order No. 3456, Contract No. MDA903-82-C-0160
Monitored by the
Defense Supply Service, Washington, D.C.

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION LIMITED

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of the Defense Advanced Research Projects Agency or the United States Government.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SAI-84-176-WA	2. GOVT ACCESSION NO. AD-A130251	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) IMAGE UNDERSTANDING Proceedings of a Workshop, June, 1983		5. TYPE OF REPORT & PERIOD COVERED ANNUAL TECHNICAL October, 1982-June, 1983
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) LEE S. BAUMANN (Ed.)		8. CONTRACT OR GRANT NUMBER(s) MDA903-82-C-0160
9. PERFORMING ORGANIZATION NAME AND ADDRESS SCIENCE APPLICATIONS, INC. 1710 Goodridge Drive, 10th Floor McLean, Virginia 22102		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA ORDER No. 3456
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209		12. REPORT DATE June, 1983
		13. NUMBER OF PAGES 345
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR RELEASE; DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Digital Image Processing; Image Understanding; Scene Analysis; Edge Detection; Image Segmentation; CCDArrays; CCD Processors

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This document contains the technical papers and outlines of semi-annual progress reports presented by the research activities in Image Understanding, sponsored by the Information Processing Techniques Office; Defense Advanced Research Projects Agency. The papers were presented at a workshop conducted on 15-16 September 1982 in Palo Alto, California.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

SECTION II - TECHNICAL REPORTS PRESENTED (Continued) PAGE

SECTION III - TECHNICAL PAPERS NOT PRESENTED

TABLE OF CONTENTS (Continued)

SECTION III - TECHNICAL PAPERS NOT PRESENTED (Continued) PAGE

SECTION III - TECHNICAL PAPERS NOT PRESENTED (Continued) PAGE

FOREWARD

The Fourteenth Image Understanding (IU) Workshop sponsored by the Defense Advanced Research Projects Agency, Information Processing Techniques Office was held in Arlington, Virginia, on June 23rd, 1983. The workshop was conducted as a full day session of the Computer Vision and Pattern Recognition Conference presented by the Computer Society of the IEEE.

Commander Ronald B. Ohlander, USN, the Intelligent Systems Program Manager for the DARPA/IPTO, welcomed the large audience consisting of research personnel involved in the Image Understanding Program, Government personnel from various departments and agencies, and attendees from the CVPR conference interested in the research efforts ongoing in this DARPA sponsored program. He noted that the existence of so large and varied a conference as the CVPR, which has covered two days of tutorials and three days of general sessions as well as this workshop, indicates the high level of interest and wide variety of mature research now ongoing in the Image Processing field. This is the second time that DARPA has coordinated its IU workshop with a professional society active in the field, remarked CDR Ohlander, the first being a joint meeting in April 1981 with the Society of Photo Optical Instrumentation Engineers (SPIE). CDR Ohlander indicated that the growing body of highly sophisticated researchers, particularly in the Universities but also in the general industrial community, was a paramount factor in the growing usefulness of IU science in both military and non military fields of endeavor. This combined meeting, he concluded, is an excellent opportunity for users and theoreticians to interact to the mutual benefit of both groups.

The morning and first part of the afternoon session of the workshop comprised thirteen technical reports. These reports were selected by the principal investigators as representing an interesting facet of their research programs. Due to the press of time, each organization involved in the program was limited to only one presentation. However, in order to provide as complete a record as possible for use of government sponsors, all reports produced by the various researchers in the DARPA Program are included in this proceedings. A few reports were presented at other sessions of the CVPR Conference and are therefore published in the CVPR proceedings as well as in this volume.

The remainder of the workshop consisted of a panel discussion on the topic of, "Most important problems to be addressed in IU over the next few years". This subject was included in order to elicit comments from the wide experience available in the audience as well as the expertise of the panel discussants.

This proceedings has been supplied to the Defense Technical Information Center (DTIC) and copies may be secured from that Agency by writing to the following address:

Defense Technical Information Center
Cameron Station, Bldg. #5
Alexandria, Virginia 22314

A small charge is assessed by the DTIC for reproduction expenses. Accession number for this proceedings is not yet available but will be assigned by the DTIC within the next thirty days. Accession number for previous issues are listed on the following page.

The materials for the cover of this proceedings were supplied by Dr. Martin Herman of Carnegie-Mellon University. Dr. Herman described the meaning of the process with this description:

> The layout shows the flow of events in the 3D Mosaic scene understanding system. The stereo aerial photographs show part of Washington, D. C. The 3D wire-frame description of the scene was produced by a process that extracted and matched junctions from the images. A geometric modelling process then converted the wire frames into a surface-based description of the scene. The reconstructed buildings

are shown in the two bottom pictures. In the picture on the lower right, gray scale obtained from one of the top images is mapped onto the faces of the buildings. The stereo reconstruction process represents one step in the 3D Mosaic system, which obtains a more complete description of the scene by incrementally accumulating information derived from multiple viewpoints. The researchers on this project include Dr. Martin Herman, Dr. Takeo Kanade, Mr. Shigeru Kuroe, and Mr. Duane Williams.

A more complete description may be found in Dr. Herman's paper, "Monocular Reconstruction of a Complex Urban Scene in the 3D MOSAIC System", reproduced in section III of this proceedings.

Mr. Tom Dickerson of Science Applications, Inc. was responsible for the artwork and lay-out for the proceedings cover. Appreciation is also due Ms. Neville Worthington of Science Applications, Inc. for her assistance with arrangements, and particularly for typing support and in putting together this proceedings. Finally, our thanks to the Computer Society, IEEE, for their cooperation and assistance during the planning and execution for the conference and workshop. Particularly helpful were Mr. Harry Hayman and Ms. Jerry Katz of IEEE and Dr. Takeo Kanade of Carnegie-Mellon University, the conference chairman.

Lee S. Baumann
Science Applications, Inc.
Workshop Organizer

DEFENSE TECHNICAL INFORMATION CENTER

ACCESSION NUMBERS FOR PREVIOUS

I.U. WORKSHOPS

ISSUE	DATE		NUMBER
APRIL	1977	ADA	052900
OCTOBER	1977	ADA	052901
MAY	1978	ADA	052902
NOVEMBER	1978	ADA	064765
APRIL	1979	ADA	069515
NOVEMBER	1979	ADA	077568
APRIL	1980	ADA	084764
APRIL	1981	ADA	098261
SEPTEMBER	1982	ADA	120072

AUTHOR INDEX

AUTHOR INDEX (Cont'd)

AUTHOR INDEX (Cont'd)

SECTION I

PROGRAM REVIEWS
BY
PRINCIPAL INVESTIGATORS

D up

Image Understanding Research at CMU

Takeo Kanade

Computer Science Department
Carnegie-Mellon University
Pittsburgh, PA 15213

The goals of Image Understanding Research at CMU have been to develop basic theory for understanding 3-dimensional shapes and to demonstrate an integrated system for photo interpretation (database and interactive/automatic image interpretation techniques). For these goals we have been working in three subareas: 1) Incremental 3D Mosaic System; 2) Theory for Shape Understanding; and 3) MAPS. This report reviews our progress since the September 1982 workshop proceedings.

1. Incremental 3D Mosaic System

The Incremental 3D Mosaic system acquires a 3D surface-based description (or model) of a complex urban scene by incrementally accumulating information derived from multiple viewpoints. Since our report in the September 1982 proceedings [Herman, Kanade, and Kuroe 82], we have made significant progress in two components of the system: the component that merges information from a new view into the current model, and the component that performs monocular analysis of an image.

As shown in Figure 1, each view of a given scene (which may be either a single image or a stereo pair) undergoes analysis which results in a 3D wire-frame description that represents portions of edges and vertices of spatial structures such as buildings. In order to update the current scene model (which has been obtained from previous views), the wire-frame description from the current view must be matched with and merged into the current model. The matching step provides the coordinate transformation from the wire frames to the model and provides corresponding edges and vertices in the two. The combined result must then be converted into a new model.

The merging step works as follows. Two objects, one in the wire-frame description and the other in the model, are merged by first merging their corresponding pairs of edges and vertices into single elements by weighted averages of their positions. Next hypothesized elements (faces, edges, or vertices) in the model that are inconsistent with modified elements are deleted. To determine whether inconsistencies exist, dependencies have been recorded for each hypothesis at the time of its creation. A hypothesis is dependent on all elements whose existence directly resulted in the creation of the hypothesis. For example, if an open polygon is completed by hypothesizing a line connecting the two end points of the chain of segments, the hypothesized line is dependent on the two end lines of the chain. If one of these lines is modified or deleted, the hypothesis must also be deleted, for the conditions under which it was created are no longer valid. After all mergings and deletions, the remaining edges and vertices in the wire-frame object are added to the model object. After this is done for all objects, Those objects which are incomplete are completed using task specific knowledge, as described in [Herman, Kanade, and Kuroe 82] [Herman, Kanade and Kuroe 83].

Herman has also been developing a monocular analysis component for the 3D Mosaic system [Herman 83] (in this volume). This component reconstructs the three-dimensional shape of a complex urban scene from a single image. His approach exploits task-specific knowledge involving block-shaped objects in an urban scene. First, linear connected structures in the image are generated: these are meant to represent edges and vertices of buildings. Next, the 2D structures are converted into 3D wire frames. Finally, a surface-based description of the scene is generated from the wire frames.

In our database, we have two different views of part of Washington, D.C.: a stereo pair for one view and a single image for the other. Eventually, we will merge the 3D wire frames obtained from the single image with the scene model obtained from the stereo pair.

2. Theory for Shape Understanding

At CMU, we have been working on the geometrical aspects of image constraints for extracting shape from images. We have continued our effort in this important area to develop fundamental theories and their applications for recovering three-dimensional shapes from images. Our new results include:

- Theory of circular straight homogeneous generalized cylinders [Shafer and Kanade 83]
- Stereo by dynamic programming in a three-dimensional search space [Ohta and Kanade 83]
- Optical flow methods for measuring object motion in an X-ray image sequence [Cornelius and Kanade 83]
- A method for obtaining topological correspondence of line drawings of multiple views [Thorpe and Shafer 83]

2.1 Theory of Generalized Cylinders for Vision

Motivated first work in the shadow analysis [Shafer and Kanade 82], in which the shadow volume is a generalized cylinder, Shafer and Kanade [Shafer and Kanade 83] (in this volume) have investigated the formal properties of generalized cylinders. In recent years, Binford's generalized cylinders have become an important tool for shape representation in image understanding systems [Brooks 81]. However, research has been hampered by a lack of analytical results for these shapes. Shafer and Kanade start with a definition for Straight Homogeneous Generalized Cylinders, those generalized cylinders with a straight axis and with cross-sections which have constant shape but vary in size. This class of shapes, while still quite large, has properties which make considerable analysis possible.

The results begin with deriving formulae for points and surface normals for these shapes. Theorems are presented concerning the conditions under which multiple descriptions can exist for a single solid shape. Then projections and contour generators are analyzed. The strongest results are obtained for solids of revolution (which are named Right Circular SHGCs), for which a closed-form method for analyzing image contours is presented. Shafer and Kanade has shown that a picture of the contours of a solid of revolution is ambiguous, with one degree of freedom related to the angle between the line of sight and the solid's axis. The ambiguity can be resolved by other constraints such as those from shadow contours.

2.2 Optical Flow Method for Object Motion in X-ray Images

In calculating optical flow from an image sequence, Horn and Schunck [Horn and Schunck 81] assumed that the image brightness corresponding to the same physical point does not change, together with the assumption of smoothness of velocity over the image. However, this assumption of zero brightness change severely limits the allowable motions. Rotations, translations in depth, and deformations often result in a change in the image brightness corresponding to a single physical point. Also, the assumptions of smoothness and zero brightness change do not hold at the boundary of the object.

It was shown that the problems of assuming zero brightness change is magnified when we try to apply the method to an x-ray image sequence. (In x-ray images, the brightness of each point depends on the amount and density of the mass between the x-ray source and the film.) Cornelius and Kanade [Cornelius and Kanade 83] (in this volume) have adapted the optical flow algorithm so that it can handle the brightness change and cope with the difficulty caused by the smoothness assumption across the boundary. This algorithm assumes: (a) the

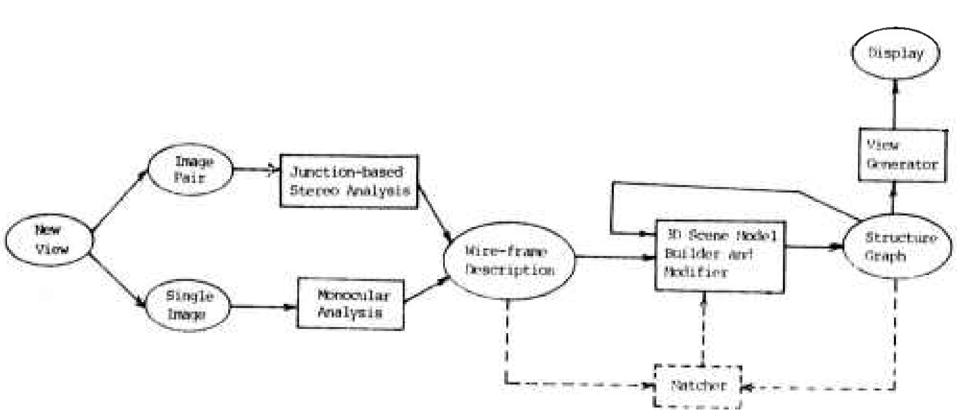


Figure 1: The current structure of the Incremental 3D Mosaic system: boxes are major modules and ellipses are data structures

brightness changes corresponding to a single physical point can be described by the first-order expansion of the image intensity function $I(x,y,t)$; (b) the velocity field (v_x, v_y) changes smoothly in a neighborhood, unless the neighborhood contains an occluding boundary; (c) the rate of change in brightness (dI/dt) is smooth in a neighborhood. An iterative procedure was devised to compute the velocity field and the change of brightness (ie., change of thickness in the case of x-ray images) under these conditions.

This algorithm can correctly recover the object motion from the x-ray images of an expanding ellipsoid. We have actually applied the method to real x-ray images of a dog's heart taken on film at 60 frames a second, in which a radio-opaque dye was injected into the pulmonary artery just before the image sequence was taken. For this case, the changes in brightness will reflect the expansion or contraction movement of the heart in the direction perpendicular to the image plane since the dye filled heart is the primary source of motion. We have generated a movie of the velocity vectors for an entire heart cycle and shown that it coincides well with the apparent motion seen in the actual cine angiogram.

2.3 Stereo by 3D Search

Ohta and Kanade [Ohta and Kanade 83] have been developing a stereo algorithm to obtain an optimal matching surface in a three dimensional search space. Their approach is purely computational. When a pair of stereo images is rectified so that the epipolar lines are horizontal scan lines, we can search for a pair of corresponding points in right and left images within the same scan lines. We call this search *intra-scanline* search. This intra-scanline search can be treated as the problem of finding a matching path on a two dimensional search plane whose axes are right and left scanlines. A dynamic programming technique can efficiently handle this search [Baker 82]. The intra-scanline search alone, however, does not take into account mutual dependency between scanlines in a image; that is, *inter-scanline* search is necessary to find the consistency across scan lines.

As shown in Figure 2, we cast the problem of stereo as that of finding a matching surface (i.e., a set of matching paths) in a three dimensional search space, which is a stack of the 2-D search planes and whose axes are left-image x position, right-image x position and the scan line (y position of image). Vertically connected edges provide the consistency constraints across the scan line axis. Thus, stereo involves two searches: one is intra-scanline search for possible correspondence and the other is inter-scanline search for consistency between connected edges. Ohta and Kanade employ dynamic programming for both searches.

The matching is based on edges, and the positions of edges are obtained as zero-crossings of the 1D Laplacian (taken along each scan line) in both left and right images. The intra-scanline search locates many partial paths for each pair of left and right scan lines, as candidates of components which may consist of the final matching surface. The inter-scanline search uses those partial paths as elements, and searches for the combination of them which is most consistent with connected edges. These two searches proceed simultaneously. The criteria (i.e., the cost function) in the search involve a monotonicity assumption, the similarity of intensity between edges, and surface smoothness.

Our main task domain is urban aerial photographs, but images in other domains are also used to show the performance of our stereo. Figure 3 is a typical example of aerial stereo images. Figure 4 (a) shows the disparity map obtained, and Figure 4 (b) shows an isometric plot of the depth map. Notice that the detailed structures of the roof of the building and the bridge over the highway are clearly extracted. The output of this stereo program will be used as another source of 3D information in the Incremental 3D Mosaic system.

3. MAPS

MAPS is a large integrated image/map database system for photo interpretation tasks. It contains high resolution aerial photographs, digitized maps and other cartographic products, combined with detailed 3D descriptions of man-made and natural features in the Washington D. C. area [McKeown and Kanade 81] [McKeown and Denlinger 82]. In the September 1982 proceedings, McKeown [McKeown 82] reported the addition of the concept map to facilitate inquiries at the symbolic level. Since then, the concept map has been used to build a hierarchy tree data structure which represents the whole-part relationships and spatial containment of map feature descriptions [McKeown 83] (in this volume). Unlike regular decomposition methods such as quad-tree organizations, the hierarchical containment tree permits a hierarchical search in the database based on natural relations among features which are intrinsic to the conceptual map and may have some analogy with how humans organize a "map in the head" to avoid search. Thus the hierarchy tree improves the speed of spatial computations by quickly constraining search to a portion of the database.

As an application of MAPS, McKeown has started investigation of rule-based systems for the control of image processing and

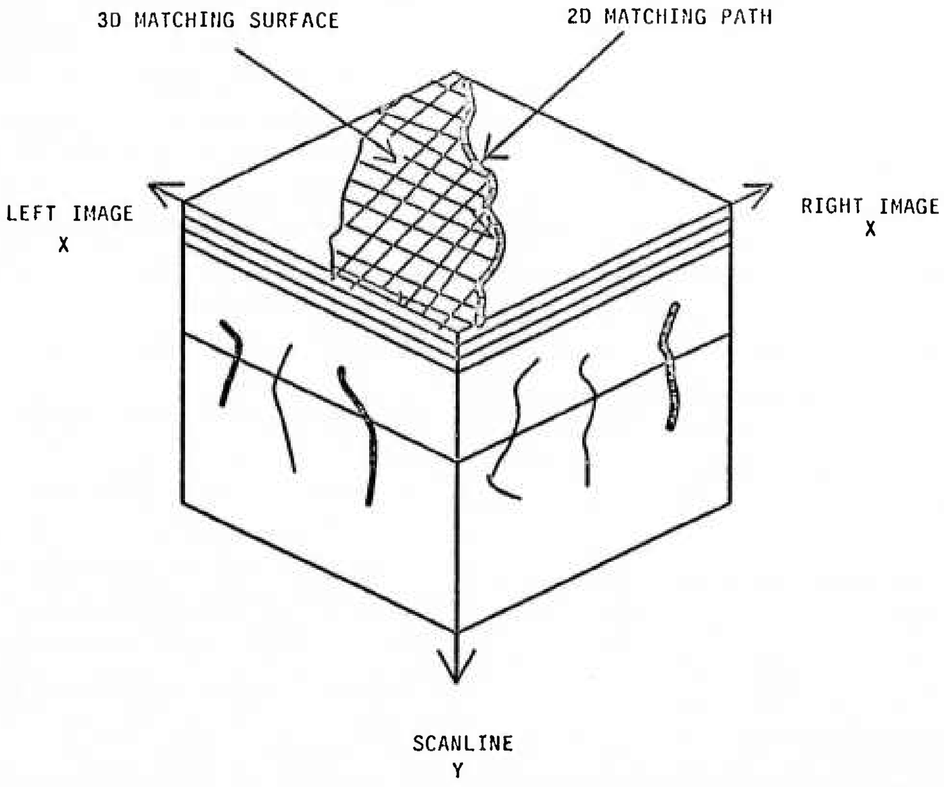


Figure 2: The three-dimensional search space for stereo matching

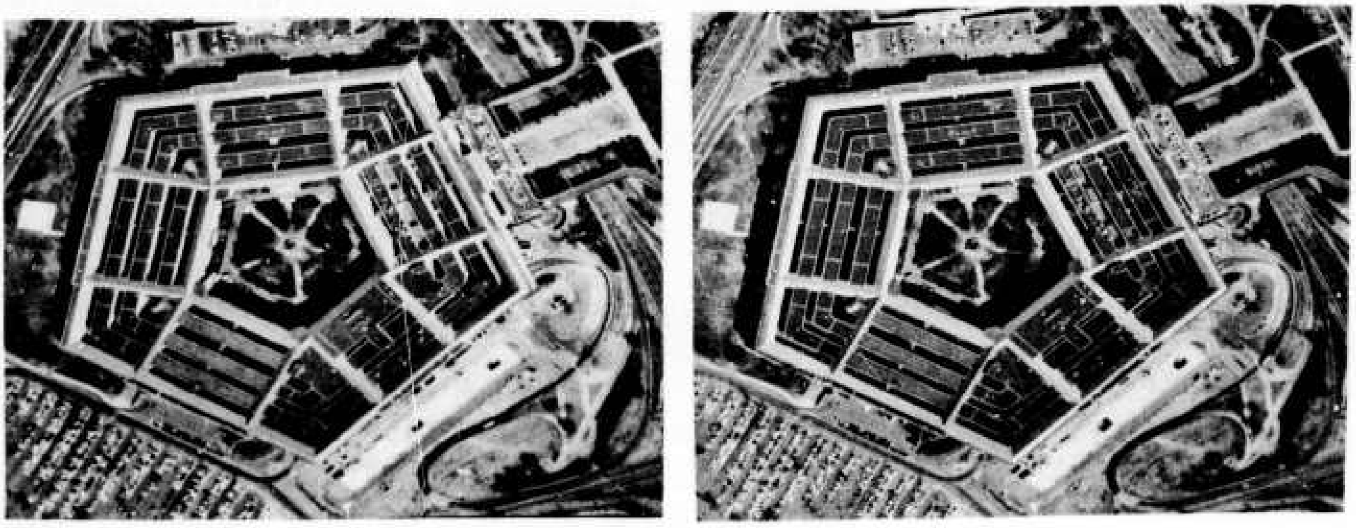
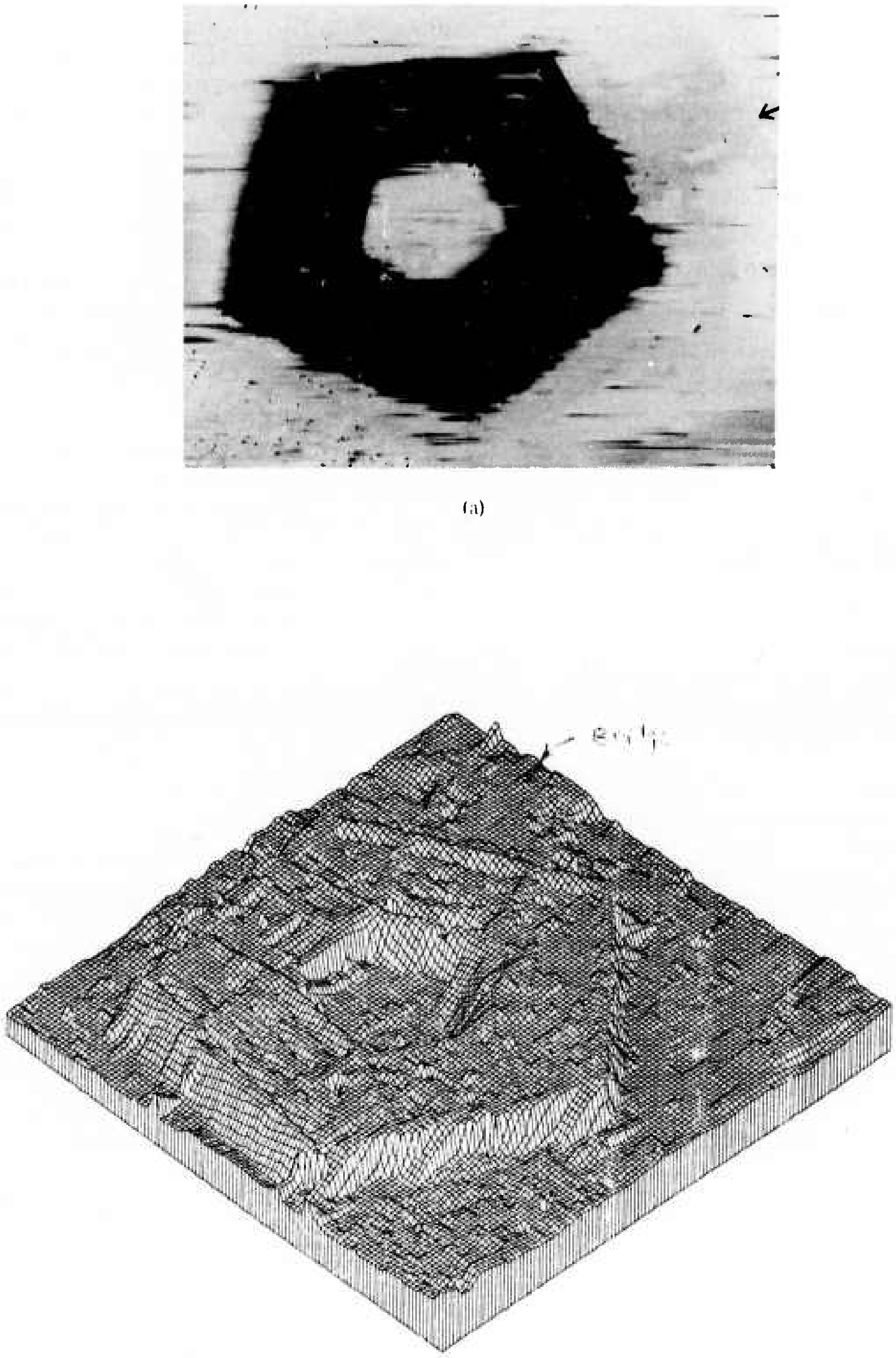


Figure 3: A example of aerial stereo images.



(a)

(b)

Figure 4: (a) Disparity map obtained; (b) Isometric plot of the depth map. Note that the detailed structures of the roof of the building and the bridge over the freeway are detected.

interpretation with respect to a world model. The SPAM system [McKeown and McDermott 83] is a system for testing the idea of using the combination of task independent low-level image processing tools, a rule-based system and a map database expert.

4. Systolic Array Processors for Vision

Together with the VLSI group of CMU, we have started investigating applications of systolic array processors made of PSCs (Programmable Systolic Chips) [Fisher et al. 83] [Fisher, et al. A 83] to image processing. Example tasks we are considering include: smoothing, edge detection, optical flow, iterative image registration, and matching by dynamic programming. We expect one to three orders of magnitude improvements in the speed of performing these image processing tasks over conventional machines.

References

[Baker 82] Baker, H. H.
Depth from Edge and Intensity Based Stereo.
Technical Report AIM-347, Stanford Artificial Intelligence Laboratory, 1982.

[Brooks 81] Brooks, R. A.
Symbolic Reasoning among 3-D Models and 2-D Images.
Artificial Intelligence 17:285-349, 1981.

[Cornelius and Kanade 83]
Cornelius, N. H. and Kanade, T.
Optical Flow Method to Measure Object Motion in Reflectance and X-Ray Image Sequences.
Technical Report (in preparation), Computer Science Department, Carnegie-Mellon University, 1983.

[Fisher et al. 83] Fisher, A.L., Kung, H.T., Monier, L.M. and Dohi, Y.
Architecture of the PSC: A Programmable Systolic Chip.
In *Proceedings of the Tenth International Symposium on Computer Architecture.* June, 1983.

[Fisher, et al. A 83]
Fisher, A.L., Kung, H.T., Monier, L.M., Walker, H. and Dohi, Y.
Design of the PSC: A Programmable Systolic Chip.
In Bryant, R. (editor), *Proceedings of the Third Caltech Conference on Very Large Scale Integration*, pages 287-302. caltech, Computer Science Press, Inc., March, 1983.

[Herman 83[Herman, M.
Monocular Reconstruction of a Complex Urban Scene in the 3D MOSAIC System.
In these proceedings , 1983.

[Herman, Kanade and Kuree 83]
Herman, M., Kanade, T., and Kuroe, S.
The 3D MOSAIC Scene Understanding System.
To apear in Proc. IJCAI-83 , August, 1983.

[Herman, Kanade, and Kuroe 82]
Herman, M., Kanade, T. and Kuroe, S.
Incremental Acquisition of a Three-dimensional Scene Model from Images.
Technical Report CMU-CS-82-139, Computer Science Department, Carnegie-Mellon University, 1982.
Also in *Proc. DARPA Image Understanding Workshop*, Sept. 1982.

[Horn and Schunck 81]
Horn, B. K. P. and Schunck, B.
Determining Optical Flow.
Artificial Intelligence 17:185-203, 1981.

[McKeown 82] McKeown, D. M. Jr.
Concept Maps.
In *Proc. DARPA Image Understanding Workshop*, pages 142-153. Sept., 1982.

[McKeown 83[McKeown, D. M.
MAPS: The Organization of a Spatial Database System Using Imagery, Terrain, and Map Data.
In these proceedings , 1983.

[McKeown and Denlinger 82]
McKeown, D. M. Jr. and Denlinger, J. L.
Graphical Tools for Interactive Image Interpretation.
In *SIGGRAPH '82 (Computer Graphics Vol. 16, No. 3)*, pages 189-198. July, 1982.

[McKeown and Kanade 81]
McKeown, D. M. Jr. and Kanade, T.
Database Support for Automated Photo Interpretation.
In *Proc. DARPA Image Understanding Workshop*, pages 7-13. April, 1981.

[McKeown and McDermott 83[
McKeown, D.M. and McDermott, J.
Toward Expert Systems for Photo interpretation.
In *IEEE Trends and Applications '83.* May, 1983.

[Ohta and Kanade 83]
Ohta, Y. and Kanade, T.
Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming.
Technical Report (in preparation), Computer Science Department, Carnegie-Mellon University, 1983.

[Shafer and Kanade 82]
Shafer, S. and Kanade, T.
Using Shadows in Finding Surface Orientations.
Technical Report CMU-CS-82-100, Carnegie-Mellon University, Jan., 1982.

[Shafer and Kanade 83]
Shafer, S. and Kanade, T.
The Theory of Straight Generalized Cylinders.
Technical Report CMU-CS-83-105, Computer Science Department, Carnegie-Mellon University, 1983.
Also in these proceedings.

[Thorpe and Shafer 83]
Thorpe, C. and Shafer, S.
Topological Correspondence in Line Drawings of Multiple Views of Objects.
Technical Report CMU-CSD-83-113, Computer Science Department, Carnegie-Mellon University, 1983.
Also to appear in *Proc. IJCAI-83.*

IMAGE UNDERSTANDING RESEARCH AT COLUMBIA

John R. Kender
Department of Computer Science, Columbia University
New York, NY 10027

Abstract

The Image Understanding Project at Columbia has centered its efforts on basic "middle-level" vision research: the representations and algorithms concerned with deriving surface information from low-level aggregate cues. At present, the effort has four major concerns: theory and analysis, integrated systems, image research aids, and high-speed hardware. This report on our first full year summarizes our progress in each of these areas.

1 Introduction

The Image Understanding Project at Columbia is new and small, but growing. In our first full year, we have acquired an operating laboratory, and defined and attacked our research concerns. (Currently, our experimental base consists of a VAX 750 with Grinnell 275, with CMU image and graphic software operating on USCIPI and other images. Additional hardware and software enhancements are planned.)

Our research emphasis is on that level of image understanding that moderates low-level cues into surface information. We have developed several new algorithms that make some of these transformations possible, and have begun to quantify their accuracy. Work is under way to integrate several of these surface-constraining algorithms into a coherent, distributed system; two separate free-running algorithms have been executed and are being refined. Because the algorithms and their control is complex, we are implementing various graphic ways in which the rich intermediate data can be represented easily to the experimenter. Lastly, we have devised and simulated some low-level vision algorithms for a novel supercomputer being independently developed at Columbia.

2 Theory and Analysis

Much of our theoretical work concerns the calculation of surface orientation constraints from low-level image cues. One representation that has proven very useful for this and other tasks is the gradient space--independently of whether the image is taken under orthographic or central projection. We have helped to summarize some of its most salient properties (especially those under projection) in a type of researcher's reference card [Shafer 83; Shafer 82]. We have also highlighted some of the difficulties that can occur under perspective; algorithms known for their utility under orthography can fail in unexpected ways [Kender 82a].

Many of the algorithms we have devised for our middle-level work are derived from a central methodological paradigm called "shape from texture" [Kanade 83; Kender 82b]. We have now applied the paradigm in two additional areas, deriving additional surface constraint relations and procedures. (Versions of these two papers appear in this proceedings.)

The first area concerns gravity, which induces certain preferred scene orientations. We have shown how gravitationally-related labels such as "vertical" can be used in the gradient space, and how such knowledge can generate additional constraints on surfaces [Kender 83a; Kender 83b]. In particular, we have shown that sensor parameters, surface parameters, and environmental labels mutually interact so that knowledge of any two constrains the third; further, often this knowledge can be heuristically derived using Hough-like methods.

The second area concerns linear extents: image primitives that possess measurable length. We have shown how assumptions of equality of extent provide surfaces constraints, sometimes in non-intuitive ways [Kender 83c]. In particular, under orthography, lengths behave very much like right angles; under perspective, certain configurations induce several simple iconic (image plane) geometric constructions for vanishing points.

Lastly, we (David Lee) have initiated the analysis of the error behavior of a few of these algorithms. We believe that a fruitful framework is that of the information-centered approach under independent development at Columbia. We expect to be able, given a desired accuracy of surface orientation, to derive lower limits on the resolution necessary in the image, or on the confidences necessary in the image primitive array.

3 Integrated Systems

We (Mark Moerdler) have started work on the design and implementation of a middle-level vision system that integrates knowledge about surfaces from multiple independent sources. Present design is patterned on the blackboard model of perceptive systems. Each source derives surface information on the basis of one particular shape algorithm.

Two such sources have been coded. Although primitive and under refinement, their results are shown in the figures following this report. Figure 1 shows a synthetic image ("Manhattan Sunrise") with two surfaces sharing a common orientation; the lower surface is composed of two textures. In Figure 2, an algorithm based on equal extents, applied to the "waves", generates multiple vanishing points, very near the actual (but invisible) vanishing line. In Figure 3, an algorithm based on the detection of colinearities in random textures (Peter Wesley), applied to the "sand", generates a smear of vanishing points that straddles the vanishing line. Since vanishing lines map one-to-one into surface orientations, these two algorithms implicitly calculate local slant and tilt.

4 Image Research Aids

One problem with the development of image understanding systems is the vast amount of complex intermediate data that they produce. In particular, the middle levels of vision are replete with partial assertions about the underlying surfaces.

Since surfaces have two parameters of orientation and one parameter of depth, and since each image point may have multiple surface hypotheses, the problem of observing and understanding an executing system becomes one of human-compatible graphic economy.

We (Paul Douglas) have begun research into the various modalities of human vision that can be exploited in this task. Primarily, we are constructing a surface synthesis system that will artificially texture (locally planar) regions of an image in ways that suggest their orientations. Additionally, we have begun to explore the ways in which orientation uncertainty and/or constraints can be graphically displayed by means of icons, motion, or color. Our initial icons are based on "sequins" (circles seen in perspective).

5 High-speed Hardware

Several parallel machine architectures have been proposed that perform image understanding algorithms at high speed. The NON-VON supercomputer being built at Columbia is a tree-structured one. Its primary processing system consists of a very large number of very small processing elements (PEs), each containing a small amount of RAM and some hardware for performing arithmetic and logical operations. The PEs are connected together in the form of a complete binary tree. We (Hussein Ibrahim) have found that this architecture lends itself easily and naturally to the representation and manipulation of binary images by quad trees.

A binary picture at its finest resolution is stored in the leaves of the tree, with each PE holding one picture point. Higher levels in the tree represent coarser resolutions; building the quad tree can be done in logarithmic time. Connected components can be found in time proportional to the number of nodes actually representing regions in the tree. Several other algorithms for region properties again take logarithmic time. These algorithms have all been tested on a simulator. We expect to develop the usual complement of image processing routines, with a target task in mind.

Acknowledgements

Graduate Fellow contributors to the Image Understanding Project include Paul Douglas, Hussein Ibrahim, David Lee, and Mark Moerdler. Dan Sabbah (IBM, Yorktown) and Peter Selfridge (Bell Labs, Holmdel), are contributing Research Computer Scientists. The Vision Lab thanks CMU for its low-level image buffering and graphics processing software, which has considerably eased our systems initialization.

References

[Kanade 83] Kanade, T., and Kender, J.R. Mapping Image Properties into Shape Constraints: Skewed Symmetry, Affine-Transformable Patterns, and the Shape-from-Texture Paradigm. In *Human and Machine Vision*, Rosenfeld, A., and Beck, J., Eds., Academic Press, 1983.

[Kender 82a] Kender, J.R. Why Perspective is Difficult: How Two Algorithms Fail. Proceedings of the National Conference on Artificial Intelligence, Aug., 1982, pp. 9-12.

[Kender 82b] Kender, J.R. A Computational Paradigm for Deriving Local Surface Orientation from Local Textural Properties. Proceedings of the IEEE Computer Society Workshop on Computer Vision: Representation and Control, Aug., 1982, pp. 143-152.

[Kender 83a] Kender, J.R. Environmental Labelings in Low-Level Image Understanding. Proceedings of the 1983 International Joint Conference on Artificial Intelligence, Aug., 1983, pp. .

[Kender 83b] Kender, J.R. Environmental Relations in Image Understanding: The Force of Gravity. Department of Computer Science, Columbia University, Jan., 1983.

[Kender 83c] Kender, J.R. Surface Constraints from Linear Extents. Department of Computer Science, Columbia University, March, 1983.

[Shafer 82] Shafer, S.A., Kanade, T., and Kender, J.R. Gradient Space under Orthography and Perspective. Proceedings of the IEEE Computer Society Workshop on Computer Vision: Representation and Control, Aug., 1982, pp. 26-34.

[Shafer 83] Shafer, S.A., Kanade, T., and Kender, J.R. "Gradient Space under Orthography and Perspective." *Computer Graphics and Image Processing* (To appear 1983). Also available as CMU Technical Report CMU-CS-82-123, May, 1983

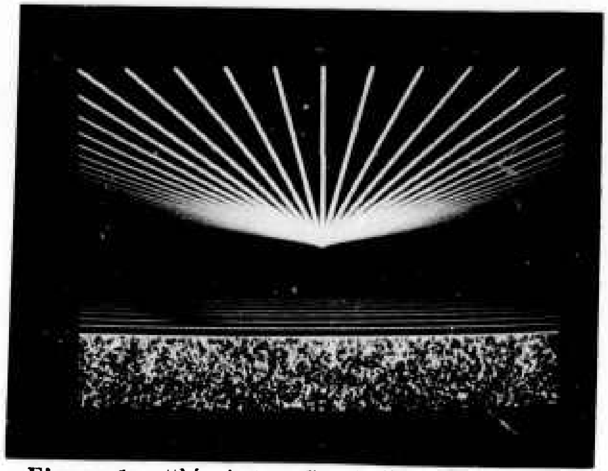


Figure 1: "Manhattan Sunrise" synthetic image.

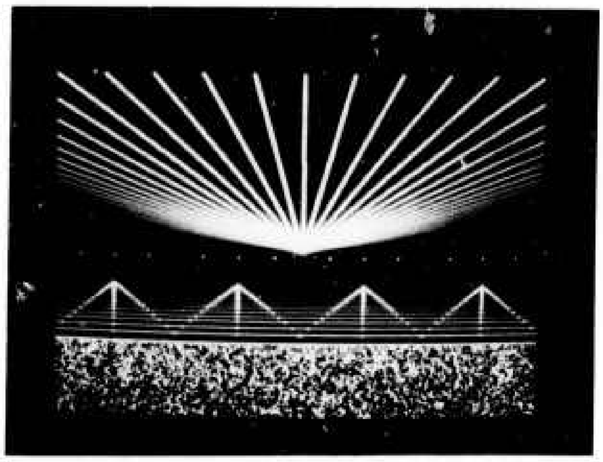


Figure 2: Equal extent method applied to waves.

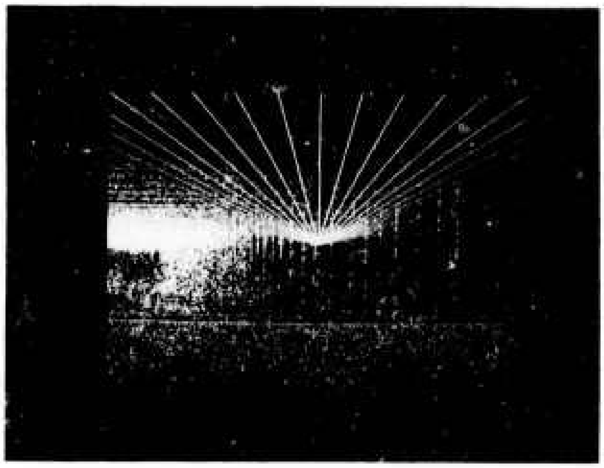


Figure 3: Colinearity method applied to image; leftmost vertical quarter only.

MIT PROGRESS IN UNDERSTANDING IMAGES

T. Poggio, S. Ullman and the staff

The Artificial Intelligence Laboratory, Massachusetts Institute of Technology

The authors' *Our overall approach to the study of vision is based on a number of representations of the visible world, reviewed in previous Image Understanding Proceedings.* Their *Our work to date has concentrated primarily on the initial representations such as the primal sketch and reflectance maps, and the computation from them of depth, surface orientations, and material properties.* The *Our current emphasis is on the integration of the different sources of information, the analysis and representation of shape, the refinement and evaluation of the individual modules, the extension of* their *our approach to deal with time varying images and moving objects, and the transfer of our results to real time hardware implementation. In this report* they *we review* their *our recent work on the analysis of edge detection, the measurement of visual motion, the correspondence problem, the refinement and evaluation of stereo algorithms, the detection of depth discontinuities, the integration of surface maps, and the interpretation of shape from contours, and the acquisition of objects with photometric stereo.*

1. Edge detection analysis

Much of our work on edge detection, discussed in previous Image Understanding Workshops, used the zero-crossing contours in the image filtered through $\nabla^2 G$ filters of different sizes. Any edge detector scheme to be used in practical applications must show considerable robustness and immunity to various types of noise. Continuing his work aimed at developing a practical real time stereo-matching system, Nishihara has examined recently the effect of image noise on the $\nabla^2 G$ convolution and the zero-crossing contours. In parallel with the effort of developing further our standard edge detection techniques and improving their reliability, we are also pursuing new approaches to the edge detection problem. In particular, we are developing, implementing and testing a new line finder . In another investigation we are characterizing general properties of edge detection schemes. We have also established some results connecting the locations of zero-crossings with the principle lines of curvature of a surface. We now review each of these four topics in turn.

Noise Sensitivity of Zero-crossings

Distortions due to noise can be considered as perturbations of the shapes of regions of constant sign in the convolution output. Zero-crossing patterns are generally stable in the presence of low to moderate image noise levels.

The most common serious distortion of these patterns—for stereo matching—occurs when two adjacent regions of constant sign merge or a single region splits as a function of noise introduced by the cameras or changing camera position.

Only a small number of pixels need change sign at strategic locations in order for such merges and divisions to occur, resulting in a large scale change of the zero-crossing geometry. The frequency of these changes is low in a high quality image, but they cannot be avoided when noise is present and contrast is low, a ubiquitous phenomenon in practical images. This distortion turns out, however, to be strongly confined to specific spatial neighborhoods of the image where the convolution magnitude is small. Outside these neighborhoods, the convolution sign is constant and stable, even for relatively large noise levels. The sign-representation dual of the zero-crossing also promises to yield more easily to a careful statistical analysis. Nishihara is investigating ways in which the approach can be used to improve noise tolerance in stereo matching [Nishihara, 1982, 1983].

Optimal edge detection operators

Canny [1983] has investigated the problem of deriving an optimal edge detection operator from a precise formulation of detection and localization [Binford 1981]. He finds that the optimal shape is (approximately) the first derivative of a Gaussian. An important property of an edge detector is that it should produce edge tokens that are accurately located. It should also have a low probability of misclassification of edges (i.e. it should produce few erroneous edges and still be able to detect weak or noisy edges). In particular, the operator should not produce multiple responses to a single edge. The ability to correctly classify potential edge points relates directly to the signal-to-noise ratio of the output of the operator, which is frequently used as the design criterion for an optimal detector. The localizing ability of the edge detector is often either ignored or only indirectly treated.

Canny's derivation consists of three steps. First, the design is constrained to linear operators only. Second, the optimal linear operators are combined in a non-linear way that is again optimal (or near optimal) with respect to the criteria of detection and localization. Finally, the edge points output from the non-linear detector are processed by a line-following procedure which assigns labels to the segments of contour and to each segment a set of parameters that describe the type of edge transition (amplitude of the step, uncertainty in amplitude, uncertainty in position). The resulting operators have been implemented in microcode on a LISP machine, and form the basis for our work on smoothed local symmetries and shape from contour. The operator has also been applied to textured images to generate hierarchical texture descriptions.

The linear operator is directly optimized with respect to both signal-to-noise ratio and localization. Canny shows that there is an uncertainty principle relating the two quantities and that, because of noise, an edge cannot be simultaneously detected and localized with arbitrary precision. There is a unique operator shape (approximately the first derivative of a Gaussian) that attains this limit. The width of the operator determines the tradeoff in output signal-to-noise ratio versus localization. A narrow operator gives better localization but poorer signal to noise ratio and vice-versa. To handle variations in the signal to noise ratio in the image, operators of several widths are used. Where several operators respond to the same edge, one of them is selected by the algorithm so as to give the best localization while preserving an acceptable signal-to-noise ratio. When the one dimensional formulation is extended to two dimensions, the same criteria of optimality are used. This leads to a system of directional operators, with their noise estimation and edge detection all being performed independently.

The automatic switching between operators requires local estimation of the noise energy in the operator outputs. This is difficult because there is little information available at the operator outputs to indicate whether a response is due to an edge or to noise. Canny has developed a scheme that uses a model of an edge (in this case a step edge) to predict the response of each operator. He then removes responses of this type to leave the response due to noise alone. The noise estimation is done from the outputs of the operators rather than directly from the image, because detection and localization performance is determined by that component of the image noise parallel to the operator direction, and which lies within the bandwidth of the operator. Where image noise is not spectrally flat, and in particular where there is fine texture (element size much smaller than the operator width), the texture may be modelled as directional noise, and the detector will still be able to respond to weak edges in directions where there is little texture energy.

The detector is being evaluated in comparison with several other well-known detectors, such as the Marr-Hildreth Laplacian of Gaussian operator (1980) and the second directional derivative detector of Haralick (1982). Experiments are being performed using the operator as the front end for the Marr-Poggio stereo algorithm (Grimson 1981a,b) as well as subjective evaluations of the detector output on a variety of natural images, in particular on images that contain boundaries between textured regions. The multiplicity of operators enables the detector to locate intensity changes that are occurring at different scales in the image. The use of directional operators allows it to find weak linear edges when the signal to noise ratio is very poor. It is felt that linear edges form an important subclass of intensity changes and that they occur often enough in real images to warrant special treatment. The traditional problems with highly directional operators were that they tended to extend the boundaries of objects beyond corners and gave polygonal responses to curved surfaces. These are dealt with in the new detector by the addition of applicability constraints for each directional operator based on how well the image locally approximates a linear edge.

The detector has also been used as the front end for two hand-eye vision programs. The first of these simply tracks contours drawn on some surface. The second takes the raw edges that mark the boundaries of objects and produces bounding polyhedra of minimum additional area. The latter will be used in conjunction with automatic path planning programs.

Parallel to Canny's development of an optimal edge detector, Poggio and Torre have begun an investigation, presently in progress, of edge detection by dividing the problem into two main steps: a derivative operation and a filtering operation to reduce the noise. Each of these steps can be characterized in general terms. If the detection of edges is based on detection of extrema in the output of the filter then directional derivatives should be used in connection with directional odd filter functions. If edge detection is to be performed *via* zero-crossing detection then rotationally symmetric differential operators must be used together with symmetric filter functions. If the differential operator is linear the two steps of differentiation and filtering commute and associate with interesting implications for fast hardware. For nonlinear differential operators the two operations in general must be performed separately and furthermore their order is important. Poggio and Torre have examined in particular two rotationally symmetric operators : the second directional derivative along the gradient – a non-linear operator – and the Laplacian – a linear operator. It is easy to show that there are edges that escape detection by the Laplacian but not by the second derivative along the gradient. Furthermore, the zero-crossings of the Laplacian coincide with the zero-crossings of the second directional derivative along the gradient if, and only if, the mean curvature of the intensity function is locally zero.

Three classes of filters have been analyzed in detail: bandlimited, support limited and filters with minimal uncertainty in space and frequency. The filters of the first class can be synthetized in terms of linear and circular prolate functions; in the second class, Haar functions are the most interesting basis for optimal filters; the third class leads to the study of Hermite functions. Poggio and Torre derive formulae for computing the uncertainty of an arbitrary filter using its decomposition in Hermite functions. They also observe that a filter of minimal uncertainty combines maximum localization in space with a minimum number of zeros in its output to Gaussian white noise. In particular, the second derivative along the gradient, successively smoothed by a circularly symmetric Gaussian filter is a near-optimal scheme in terms of these criteria. In a separate investigation, we report on a 2-D version of Logan's theorem, which gives sufficient conditions for the completeness of the zero-crossing representation in the case of directional bandpass filters [Poggio et al., 1982].

Lines of curvature and zero-crossings

In recent years, workers in vision have shown considerable interest in the principal lines of curvature of surfaces. For example, curvature patches have been proposed as a representation for visible surfaces [Brady 1983] and there exist various schemes for dividing objects into parts based on extrema and zeros of curvature [Brady 1983, Hollerbach 1975]. There is also some evidence from line drawings [Stevens 1981] that curves in an image are interpreted as lines of curvature. However, it has been suggested that the principal lines of curvature of a surface can only be computed indirectly and with great difficulty. The complexity of the calculations also implies poor numerical behaviour and excessive sensitivity to noise.

Yuille [1983] proves some results about zero crossings and the principal lines of curvature of a surface. He relates the image to the underlying surface geometry by the image irradiance equation [Horn 1977] and suggests that the principal lines of curvature can be computed *directly* from the image.

Various directional zero crossing operators are considered. It is shown that directional zero crossings do not necessily correspond to physical zero crossings (i.e., those that correspond to sharp changes in the image irradiance). A result is derived that implies that directional zero crossings are physical only if their direction is along the line of greatest change of the image irradiance. Such directional operators have been argued for by Canny [1983] and Poggio and Torre [see Poggio, 1982, 1983]. Conversely, a probabilistic argument shows that the directions of greatest change of the image irradiance are most likely to be along the lines of principal curvature. This suggests that many, if not most, of the physical zero crossings are directional zero crossings along the principal lines of curvature.

Finally, Yuille proves some results about the distribution of zero crossings along lines of curvature. The starting point is the work of Grimson on surface consistency [Grimson 1981b]. With relatively weak assumptions about the reflectance function, Grimson derived neccessary and sufficient conditions in one dimension for the occurence of directional zero crossings in the image irradiance in terms of the surface geometry. He then used some probabilistic assumptions about the reflectance surface to extend this result to two dimensions and prove the Surface Consistency Theorem. This theorem was the basis for his theory of surface interpolation.

Yuille shows, without any probabilistic assumptions, that Grimson's result can be generalized to give necessary and sufficient conditions for the occurence of directional zero crossings along the principal lines of curvature. We call this result the *Line of Curvature Theorem*. It suggests that many, if not most, of the physical zero crossings can be associated with points on the lines of principal curvature which are near the extrema of the principal curvatures. This supports the view that lines of principal curvature can be computed directly from the image. In turn it supports the curvature patch representation.

2. The computation of visual motion

In the area of visual motion analysis, Hildreth and Ullman have explored a zero-crossing based approach to the computation of the two-dimensional velocity field from the changing image [Hildreth & Ullman, 1982; Hildreth, 1982, 1983; Ullman & Hildreth, 1983]. The starting point was the work of Marr and Ullman (1981), in which the initial detection of motion takes place at the location of zero-crossings in the output of the convolution of the image with a $\nabla^2 G$ operator. The main computational reason for restricting initial motion measurements to the zero-crossings is that they correspond to locations in the image for which the gradient of intensity is locally maximum, and hence yield the most reliable motion measurements [Hildreth, in press]. Hildreth and Ullman have extended the work of Marr and Ullman, to allow for the computation of the projected two-dimensional velocity field that results from the general motion of three-dimensional surfaces in space.

Due to the aperture problem, local measurements of movement in the changing image only provide the component of velocity in the direction perpendicular to the local orientation of a zero-crossing contour. In particular, let $\mathrm{V}(s)$ denote the velocity field along a contour (s denotes arclength). $\mathrm{V}(s)$ can be decomposed into components perpendicular and tangent to the curve:

$$\mathrm{V}(s) = v^{\perp}(s)\mathbf{u}^{\perp}(s) + v^{\top}(s)\mathbf{u}^{\top}(s)$$

$\mathbf{u}^{\perp}(s)$ and $\mathbf{u}^{\top}(s)$ are unit direction vectors perpendicular and tangent to the contour, and $v^{\perp}(s)$ and $v^{\top}(s)$ are the magnitudes of the two velocity components. The first term in the above expression can be measured directly from the changing image. The second term cannot, and must be recovered to compute the velocity field $\mathrm{V}(s)$.

The main theoretical problem for this recovery is that $\mathrm{V}(s)$ is not specified uniquely by information available in the changing image. Additional constraint is required to compute a unique velocity field. Drawing from the work of Horn and Schunck (1981) on the optical flow computation, we use an additional constraint of smoothness of the velocity field. Physical surfaces are generally smooth, compared with their distance from the viewer; under motion, they usually generate smoothly varying velocity fields. To compute a single velocity field, we find the velocity field which is consistent with the changing image, and varies the least.

Through a mathematical analysis, it was found that the above smoothness constraint can be formulated in such a way that a unique velocity field solution is guaranteed. In particular, the local change in $\mathrm{V}(s)$ is given by $\frac{\partial \mathbf{V}}{\partial s}$; a scalar measure of this change is given by its magnitude, $|\frac{\partial \mathbf{V}}{\partial s}|$. The total variation of velocity over an entire contour can be obtained by integrating this local measure over the curve. The velocity field computation then seeks the velocity field that is consistent with the changing image, and minimizes total variation in velocity along contours. It can be shown analytically, that there exists a unique velocity field that is consistent with the measurements of $v^{\perp}(s)$ obtained from the image, and that minimizes the particular measure of total variation given by: $\int|\frac{\partial \mathbf{V}}{\partial s}|^2 ds$.

There are two classes of motion for which the velocity field of least variation is the correct physical velocity field, assuming orthographic projection of the scene onto the image. The first consists of arbitrary rigid objects undergoing pure translation. The second consists of three-dimensional objects, whose edges are straight lines, undergoing rigid rotation and translation in space. For the class of smooth curves in rotation, the velocity field of least variation is, in general, not the physically correct one. However, it is often qualitatively similar. For examples in which the true and smoothest velocity fields differ significantly, it appears that the smoothest velocity field may be more consistent with human motion perception.

The velocity field computation has been implemented, using a standard iterative algorithm from mathematical programming, known as the conjugate gradient algorithm.

If there are n parameters to compute (in our case, the x and y components of velocity), this algorithm is guaranteed to converge to the final solution in at most n steps. The method has been applied to a number of images. Qualitatively, it appears to give good results for unrestricted motion. We plan to evaluate the method further on both synthetic and natural images in the near future.

To summarize, the computation of the two-dimensional velocity field consists of two main steps: (1) initial motion measurements are obtained along zero-crossing contours, and provide the component of velocity perpendicular to the contour, and (2) motion measurements are then integrated along the contours, to compute the two-dimensional velocity field $\mathbf{V}(s)$ that minimizes total variation, given by the measure: $\int |\frac{\partial \mathbf{V}}{\partial s}|^2 ds$. Formulated in this way, a projected two-dimensional velocity field can be computed for rigid and non-rigid surfaces undergoing general motion in space. The computation can be implemented with standard optimization algorithms. Computational experiments support the feasibility of this approach to motion measurement.

3. The correspondance problem

A very general approach to the correspondence problem in either stereo or motion consists of taking a large set of local measurements for each pixel of the image and matching the most similar sets between the two images. These measurements can be regarded as nonlinear functionals representing the "primitives" on which the matching process operates. Matching constraints, dictated by the specific problem, may easily ensure uniqueness of matching. Although a large set of primitives may appear rather cumbersome and difficult to compute, massive parallel processing which begins to be feasible with the new solid state technologies, makes a scheme of this type quite attractive. Furthermore, the resulting specificity of matching primitives may avoid the extended use of complex constraints which are more difficult to implement in a highly concurrent system.

The main problem is the choice of the appropriate class of functionals. Poggio has considered the abstract computational properties of a specific class of nonlinear functionals, i.e., polynomial functionals [Poggio, 1983]. For the correspondance problem in ideal noise-free and distortion-free images, a complete set of linear functionals can be proved to be sufficient: nonlinear functionals cannot improve the matching (since linear functionals separate points in a Banach space). In practice, however, the number of measurements is finite and actually relatively small; under these conditions nonlinear operators might represent more compactly the relevant information. For instance, zero-crossing maps of $\nabla^2 G$ convolved images can be considered as the output of a quadratic functional operating on the image with support equal to the underlying Gaussian. Kass and Poggio are presently exploring correspondence schemes based on sets of nonlinear functionals. This effort is motived by a recent algorithm developed by Kass to solve the correspondence problem and based on a large set of linear functionals. The algorithm is based on the paradigm of combining independent measurements. The underlying idea is that if a dozen or so independent indications of correspondence can be combined, then no single measurement need be dependable in order for the combination to be quite reliable. A set of nearly independent linear filters based on first and second derivatives of Gaussian smoothed images was used by Kass. He was able to show that a particular computation based on these measurements can reliably determine correspondence for textured images with signal to noise ratios of two or more. An algorithm performing this computation has been applied to a few natural images with encouraging results. The algorithm and its implementation are discussed in detail in these Proceedings [Kass, 1983].

4. Refinements and evaluation of stereo algorithms

In previous IU reports, we have described the theory and implementation of Marr and Poggio's theory of human stereo [Marr and Poggio, 1979; Grimson and Marr, 1979; Grimson 1980, 1981a, 1981b]. The input to the stereo matcher is obtained by convolving the left and right images with a number of Difference-of-Gaussian filters and locating the zero-crossings in each such convolution. The matching proceeds in a coarse to fine manner, finding zero-crossings of the same contrast sign and roughly the same image orientation, within a predetermined range along horizontal slices of the rectified images, based on the general distribution of zero-crossings. As a consequence of testing the algorithm on a wide range of natural images, a number of modifications to the published algorithm have been made. First, the matching of zero-crossing points independent of their local context may lead to isolated incorrect matches. In the original published algorithm, a continuity constraint is applied using statistical measurements over areas of the image. While this was demonstrated to be sufficient on a range of test images, it occasionally led to incorrect matches near surface discontinuities or occlusions. Similar to the work of Mayhew and Frisby [1981] and Baker and Binford [1981], we have developed a continuity constraint that checks for consistency along zero-crossing contours that typically correspond to a single physical edge. This constraint implicitly incorporates the zero-crossing orientation constraint, and may be considered as being equivalent to matching a zero-crossing contour from one image against an envelope about a contour in the other image. Second, we have also investigated the sensitivity of the algorithm to vertical disparity and other image distortions. We have found that there is tradeoff between the resolution of disparity information computed by the algorithm and the sensitivity of the algorithm to vertical disparity. Computational experiments on aerial photographs have led us to redefine the matching algorithm to match zero-crossings from a line in one image to zero-crossings lying within 2 or 3 lines of the corresponding line in the second image, reducing the resolution of the available disparity information, but enabling the algorithm to match rectified images containing small residual amounts of vertical disparity. As in the original algorithm, vertical disparities beyond this range are handled by explicitly changing the vertical alignment of the images. Interestingly, psychophysical data suggest that human stereopsis relies on a registration process mediated by appropriate eye movements, to correct for vertical disparities larger than about 4'-7' (Nielsen and Poggio, forthcoming). We are presently exploring in a computational analysis the properties of the registration process with the goal of implementing this stage as an integral part of our stereo algorithm. In a separate investigation, Nishihara and Poggio [1982] have found additional support for the matching primitives used in our stereo algorithms. They have shown that the sign of the convolved images or equivalently the zero-crossings, contain sufficient information for the matcher to operate successfully even in random-line stereo pairs invented by Julesz and Spivack and claimed to require the computation of vernier cues.

The main emphasis of work on the Grimson implementation of the Marr-Poggio theory in the past year has been in applying the algorithm to aerial photography. The images tested have contained a variety of scenes. Included in these are two stereo pairs of sections of the University of British Columbia, provided by the Faculty of Forestry. One is of a combination of apartment complexes and natural terrain, (including several hundred foot high Douglas firs). The second is of a hospital complex, with a variety of different sized buildings. The third pair, supplied by Boeing Corporation, is of a complex highway intersection. The fourth pair, supplied by the Defense Mapping Agency, is of natural terrain, as is the fifth pair, supplied by the Army Engineering Topographic Labs. The sixth pair, supplied by Stanford University, is the CDC synthetic images of a building complex. An informal evaluation of the results in currently underway in conjunction with ETL.

The performance of the matching algorithm can be evaluated on two grounds, matching efficiency and disparity localization. Matching efficiency refers to the actual correspondence process applied to the zero-crossings contours. While the specific numbers clearly depend on the particular structure of the images, for these types of images we typically find that on the order of 75 to 80 percent of the available zero-crossings are assigned a correspondence (and that this usually represents on the order of 10 percent of the image for normal sized DOG filters). Of these matched zero-crossings, usually on the order of 99.5 percent of them are correct, in that they are matched to the correct zero-crossing contour in the second image. Disparity localization refers to the accuracy of the disparity values associated with a match, a value that is a function of the localization accuracy of the Marr-Hildreth edge detector as well as of the matching process itself. An evaluation of the localization accuracy of the algorithm on these images is currently underway jointly with ETL.

A different algorithm, which also represents an evolution of the original stereo theory, has been developed by Nishihara with the goal of perfecting a high speed, noise tolerant stereo matcher. Specifically we are studying techniques for minimizing a matcher's sensitivity to such distortions in noisy signals as might occur in low contrast images and in applications where lower quality cameras are used. Nishihara has found that noise sensitivity can be reduced significantly by trading off resolution for reliability in much the same way that Marr and Poggio (1979) originally proposed trading off resolution for disparity range.

He has implemented a prototype matcher using these results on top of the realtime convolution hardware he developed earlier with N. Larson (Nishihara & Larson, 1981). The system currently produces a 16 by 16 array of depth measurements every 15 seconds from vidicon camera images having order 10-20 percent noise levels. The matching volume of the device is approximately a cube with depth resolution somewhat better than its present 16 by 16 spatial resolution. Conversion to microcode from lisp should allow a doubling of the resolution obtained while maintaining or reducing the matching time.

5. Integrating surface maps

Computational vision requires the construction of rich descriptions of surface shape. Marr and Nishihara's $2\frac{1}{2}$-D sketch [Marr, 1982], a viewer-centered description of the visible surfaces in a scene, is an important intermediate representation on the road to surface analysis and, ultimately, to object recognition.

In previous reports we have described work by Grimson and Terzopoulos on the interpolation of shape information in locations were it is not specified exactly by the image. In addition to extensions of the surface interpolation theory and the problem of computational efficiency, our recent effort in the recovery and representation of surface information concentrated on the problem of integrating information of surface shape from different sources. This section summarizes the work by Terzopoulos and by Grimson in this areas. The following section describes our research in a related area – the problem of detecting and dealing with discontinuities.

Current work by Terzopoulos examined four problems in the visual analysis of surfaces. The four are: (i) *the constraint integration problem;* (ii) *the discontinuity problem;* (iii) *the interpolation problem;* and (iv) *the computational efficiency problem.* Some of the work on interpolation of smooth surfaces from raw, scattered constraints on surface shape [Grimson, 1981a,b; Brady and Horn, 1983; Terzopoulos, 1982], and investigations into computational efficiency, which came to fruition in the development of an extremely efficient multilevel surface reconstruction algorithm [Terzopoulos, 1982, 1983], have been described in previous reports. We shall therefore concentrate here on recent advances in our study of the constraint integration.

Integrating Constraints from Several Visual Sources

Each visual modality constitutes a distinct source of partial information constraining surface shape. Processes such as stereopsis and analysis of motion naturally generate local depth constraints, while processes such as shape from shading, texture, and contours naturally provide local surface orientation constraints. Surface reconstruction necessitates the integration, over several sources, of these two classes of scattered constraints.

Surface reconstruction was formulated in terms of a physical model — a variational problem describing the equilibrium of a thin, flexible plate subject to constraints. It involves the following plate energy functional:

$$\mathcal{E}_p(v) = \int\int_{\Omega} \frac{1}{2}(\Delta v)^2 - (1-\sigma)\left(v_{xx}v_{yy} - v_{xy}^2\right) dx\, dy.$$

In the generalized formulation, the influence of various constraints on the plate interpolating surface is governed by additive penalty functionals [Terzopoulos, 1983b]. Depth constraints are handled by the functional

$$\mathcal{E}_d(v) = \frac{1}{2} \sum_{(x_i,y_i)\in D} \beta_{(x_i,y_i)}\left[v(x_i,y_i) - d_{(x_i,y_i)}\right]^2,$$

while orientation constraints are handled by

$$\mathcal{E}_o(v) = \frac{1}{2} \sum_{(x_i,y_i)\in P} \alpha_{p(x_i,y_i)}[v_x(x_i,y_i) - p_{(x_i,y_i)}]^2 + \\ \frac{1}{2} \sum_{(x_i,y_i)\in Q} \alpha_{q(x_i,y_i)}[v_y(x_i,y_i) - q_{(x_i,y_i)}]^2,$$

so that the total energy functional to be minimized (over an appropriate Sobolev space of admissible functions) is $\mathcal{E}(v) = \mathcal{E}_p(v) + \mathcal{E}_d(v) + \mathcal{E}_o(v)$. The reconstructed surface is the minimizing function $v = u(x,y)$ representing a thin plate surface at equilibrium, subject to the influence of either scattered depth constraints, or scattered orientation constraints, or both. In this way, all available constraints generated by various sources are employed as an integrated whole, and the reconstructed surface is the best possible in view of the available information.

To summarize, Terzopoulos' work in surface reconstruction, as described in the last report, has been successfully generalized to deal with the constraint integration problem. He is currently refining and testing his computational theory of visible-surface representations, aiming toward a more complete understanding of the structure of the $2\frac{1}{2}$-D sketch. In addition, he is exploring the applicability of techniques that have proven to be valuable in surface reconstruction, such as the finite element method and multilevel relaxation methods, to other problems in low- and intermediate-level vision, including lightness, shape from shading, and optical flow. Preliminary results are encouraging.

Combining stereo and shape–from–shading

Previous reports have described our work on surface reconstruction, mostly based on constructing complete surface representations, consistent with the image irradiance information, from stereo depth data. While acceptable surface reconstructions can be obtained strictly from depth information, it is clear that additional boundary constraints would lead to more accurate surface representations. In order to seek such additional boundary information, we have investigated the mathematical relationship between the Marr-Poggio theory of stereo and Horn's work on shape from shading. Grimson [1982b] has shown that if the reflectance map [Horn and Sjoberg 1979] is known, then given a pair of stereo matched depth contours it is possible to determine the surface normal along the depth contour. The proof suggests a technique for finding surface normals that is essentially analogous to photometric stereo, pioneered by Horn, Woodham, and Silver [1978]. Conversely, it is possible in principle to determine certain visible surface characteristics from stereo information. Suppose that the reflectance map is of the form

$$R(\check{n}) = \rho\Big[(1-\alpha)(\check{n}-\check{s}) + \alpha(\check{n}-\check{h})^k\Big],$$

where ρ is the albedo, α determines the convex combination of the specular and matte components of the reflectance, and k is the degree of specularity. Provided one can identify points of high curvature along the zero-crossing contours, it is possible to determine the values of the parameters k, ρ and α for the corresponding portion of the image (since the values could change with changing surface material). Using an interocular separation consistent with the separation of human eyes, the technique is most effective at a distance of about one meter. The technique may find application to wide angle stereo, however, where the numerical stability of the algorithm is expected to increase.

6. Finding Discontinuities

The geometric properties of surfaces are almost certain to be discontinuous at certain locations in the scene. Depth discontinuities occur along occluding contours, while orientation discontinuities occur along surface creases. Discontinuities in surface geometry are usually, but not always, reflected in image intensities. Terzopoulos [1983a] decomposes the discontinuity problem into three subproblems: (i) the detection of discontinuities in surface geometry, (ii) the explicit representation of these discontinuities, and (iii) a characterization of their influence on visible surface reconstruction.

He argues that the first subproblem has a widespread basis in early visual processing. The detection of discontinuities is certain to require the conjunction of simultaneous events in several visual modalities; for example, the coincidence of texture boundaries or motion boundaries with sudden disparity changes. If early visual processes are made sensitive to such events, many prominent discontinuities in surface geometry may be hypothesized before surface reconstruction begins. On the other hand, discontinuities which are subtle or hidden in the primal sketch, such as those which typically occur in random dot stereograms must await detection until the surface reconstruction stage, when a full depth map becomes available. Terzopoulos has experimented with a simple method for detecting and localizing depth discontinuities during the surface reconstruction process. Localization involves finding inflections in the bending moments of the plate interpolating surface, while detection relies on the occurrence of significant disparity gradients. The method may be conceptualized as a type of edge detection over a tentative, dense depth map, and it amounts to thresholding according to the magnitude of the surface gradient at zero crossings of the Laplacian of the surface. Constraints on binocular imaging geometry can dictate appropriate bounds on the threshold.

The thin plate surface reconstruction model also suggested how to apply the finite element method to appropriately inhibit surface interpolation across discontinuites, once they have been made explicit. In particular, the surface reconstruction algorithm was generalized to handle depth discontinuities (i.e., occluding contours) and surface orientation discontinuities (i.e., creases). Generalization involves "breaking" the interpolating plate along depth discontinuities and "joining" plate patches by strips of membrane along orientation discontinuities, thus reconstructing piecewise smooth surfaces. [The mathematical details are presented in Terzopoulos, 1983b]. Once discontinuities have been detected, say, by the method described in the preceding paragraph, the reconstructed surface may be improved by a few additional relaxation iterations.

7. Shape description

Smoothed local symmetries

The description of two- and three-dimensional shape is crucial for recognition. Brady [1982a, 1982b] has developed a representation of two-dimensional shapes that combines certain features of two-dimensional projections of generalized cylinders [Nevatia and Binford 1977, Brooks 1981] and the symmetric axis transform (SAT) [Blum and Nagel 1978]. The representation has been applied to determine where to choose grasp points on a lamina for a two-fingered robot hand.

The smoothed local symmetries representation has four components. First, local symmetry is defined in a way that differs from that implicit in the SAT. Second, axes that are smooth loci of local symmetries are computed. In this way, smoothness of axes is made explicit, rather than being left implicit as in the symmetric axis transform. Third, axes whose region of support is wholly subsumed by the support of some other axis are deleted. The resulting *smoothed local symmetries* are given a parametric description called a frame. Finally, a shape is decomposed into sub-objects for which smoothed local symmetry descriptions are computed individually. The axes act as local coordinate frames and constrain the generation of descriptions of an entire shape by combining the descriptions of subshapes.

A pilot implementation of smoothed local symmetries was reported in [Brady 1982c]. It repeatedly used an algorithm, based on the mean value theorem, for determining the points at which a line entering the shape at a given orientation to the tangent emerges from the shape. In this way the local symmetries at a point could be found iteratively. The pilot implementation worked well, but was very slow.

Recently Asada and Brady [1983] have developed an algorithm that computes an approximation to the smoothed local symmetries of a shape. First, a set of feature points are computed on the shapes bounding contour, as found by the Canny edge detector. The feature points are points of high curvature or points of inflexion, and they are found by a process analogous to edge finding but applied to the orientation of the curve (a one-dimensional function of arclength). Features analogous to those computed for the original primal sketch [Marr 1976] are extracted and interpreted. Second, the shape is approximated by best fitting straight lines and circles to the feature points found in the first stage. Asada and Brady have worked out the smoothed local symmetries generated by two contours of constant curvature, and these are fit to the segments produced in the second stage. Finally, the smoothed local symmetries are used to match a database of shape models for recognition and inspection. Heide and Brady have developed a sampling algorithm for computing the smoothed local symmetries of a shape. The main emphasis of their work is developing algorithms for removing locally plausible axes that are of minor significance globally.

Bagley and Brady [1983] generate elaborate shape descriptions using a hierarchy of shape models incorporating general geometric knowledge and, at a higher level, application-specific information. These models combined with concavities in the boundary allow isolation of subshapes. They associate with each subshape a local reference frame to characterize the joining of subshapes and to help choose among multiple interpretations.

The computation of shape from contour

An important goal of early vision is the computation of a representation of the orientation of visible surfaces. Many processes contribute to achieving this goal, stereopsis and structure-from-motion being the most studied in image understanding. Three other important contributing processes are shape-from-contour, shape-from-texture-gradients, and shape-from-shading. Several psychophysical demonstrations show that shape-from-contour is significantly more powerful than shape-from-texture-gradients. Similarly, Barrow and Tenenbaum [1981, Figure 1.3 ff] suggest that shape-from-contour is a more effective clue to shape than shape-from-shading.

Brady and Yuille have investigated the computation of shape-from-contour. Many shapes are perceived as images of surfaces which are oriented out of the picture plane. Slant judgements are not determined by familiarity with contours, but on more general knowledge of shapes and surfaces. The method proposed by Brady and Yuille is based on such general knowledge, namely a preference for symmetric, or at least compact, surfaces. Note that the contour does not need to be closed in order to be interpreted as oriented out of the image plane. In general, contours are interpreted as curved three-dimensional surfaces.

Brady and Yuille develop an extremum principle for determining three-dimensional surface orientation from a two-dimensional contour. Initially, they work out the extremum principle for contours that are closed and that are assumed *a priori* to be the images of planar surfaces. They discuss how to extend this approach to open contours and how to interpret contours as curved surfaces.

The extremum principle maximizes a familiar measure of the compactness or symmetry of an oriented surface, namely the ratio of the area to the square of the perimeter. It is shown that this measure is at the heart of the maximum likelihood approach to shape-from-contour developed by Witkin [1981] and Davis, Janos, and Dunn [1982]. The maximum likelihood approach has had some success interpreting irregularly shaped objects. However, the method is ineffective when the distribution of image tangents is not random, as is the case, for example, when the image is a regular shape, such as an ellipse or a parallelogram. The extremum principle interprets regular figures correctly. Brady and Yuille show that the maximum likelihood method approximates the extremum principle for irregular figures; but that the maximum likelihood method does not compute the correct slant for an ellipse. Witkin [1981, Figure 5] provides empirical evidence that the maximum likelihood method computes a good approximation to the perceived tilt but *underestimates* the slant. Brady and Yuille prove that the maximum likelihood method consistently *overestimates* the slant of an ellipse. A more thorough investigation of the difference between the Extremum Principle and the Maximum Likelihood method is needed.

Kanade [1981, page 424] has suggested a method for determining the three-dimensional orientation of skew-symmetric figures, under the "heuristic assumption" that such figures are interpreted as oriented real symmetries. Brady and Yuille prove that the extremum principle necessarily interprets skew symmetries as oriented real symmetries, thus dispensing with the need for any heuristic assumption to that effect. Kanade shows that there is a one-parameter family of possible orientations of a skew-symmetric figure, forming a hyperbola in gradient space. He suggests that the minimum slant member of the one-parameter family is perceived. In the special case of a real symmetry, Kanade's suggestion implies that symmetric shapes are perceived as lying in the image plane, that is having zero slant. It is clear from the example of an ellipse that this is not correct. Our method interprets real symmetries correctly.

8. Object acquisition and shape from shading

Photometric stereo as developed by Horn, Woodham and Silver [Horn, et. al., 1978; Woodham, 1981] provides shape and surface orientation from multiple images of the same scene, taken under different conditions of incident illumination.

Suppose two images are obtained by varying the direction of the incident illumination. Each picture element in the two images corresponds to the same physical point, since the imaging geometry remains unchanged. The reflectance map is changed, however, and the two values for each point can determine the surface orientation. (Three views provide complete disambignation in all cases.) Photometric stereo can be implemented very efficiently in terms of a look up table set up in an initial calibration phase in which an object of known shape is imaged under the different lighting conditions. Recently, Ikeuchi and Horn have applied this technique to the difficult problem of bin picking.

One of the remaining obstacles to the widespread application of industrial robots is their inability to deal with parts that are not precisely positioned. Present methods for automating assembly operations require separate feeding of the parts, with position and attitude carefully controlled. Ikeuchi and Horn have demonstrated a system for automatically directing a mechanical manipulator to pick one object at a time out of a pile. The attitude of the object to be picked up is found using a histogram of the orientations of visible surface patches. Surface orientation is determined using photometric stereo applied to multiple images, taken with differing lighting. The resulting needle map, giving the orientations of surface patches, is used to create an orientation histogram which is a discrete approximation to the extended Gaussian image. This is then matched against a synthetic orientation histogram obtained from protoypical models of the objects to be manipulated. Such models may be obtained from CAD databases.

The system uses stored models of the objects and can identify which of several parts is seen. The output of this process has been used to direct a mechanical arm to pick up the part. The method is not restricted to cylindrical parts or even solids of revolution. Extended light sources can be used in arbitrary positions and the objects need not be restricted to ones having particularly favourable reflective properties. As we mentioned, the system adapts to these two variables using a calibration object of known shape. A second calibration step is needed to determine the transformation between the coordinate system of the manipulator and that of the camera. This type of approach may prove very useful in practical applications. We now plan to explore the potential advantages of coupling photometric stereo with Nishiahra's stereoalgorithm discussed in Section 3.

REFERENCES

Asada, H., Brady, M. "Shape descriptions for inspection". *Proc. 1st Int. Symp. Robotics Res.*, New Hampshire, August, 1983 (in prep.)

Baker, H., Binford, T. O. "Depth from edge and intensity based stereo". *Int. Jt. Conf. Art. Intell.*, **6**, 1981.

Bagley, S., Brady, J. M. "Two-dimensional shape descriptions". Submitted to AAAI Conf., Washington, D. C., 1983.

Barrow, H. G., Tenenbaum, J. M. "Interpreting line drawings as three-dimensional surfaces". *Art. Intell.*, **17**, 75-117, 1981.

Binford, T. O. "Inferring surfaces from images". *Art. Intell.*, **17**, 205-245, 1981.

Blum, H, Nagel, R. N. "Shape description using weighted symmetric axis features". *Pattern Recog.*, **10**, 167-180, 1978.

Brady, J. M. "Criteria for representation of shape". In: **Human & Machine Vision**, Rosenfeld and Beck, eds., Academic Press, 1982a.

Brady, J. M. "Parts description and acquisition using vision". *Proc. SPIE*, Washington, D.C., 1982b.

Brady, J. M. "Smoothed local symmetries and local frame propagation". *Proc. Pattern Rec. & Image Proc.*, Los Vegas, June, 1982c.

Brady, J. M., Grimson, W. E. L. "The perception of subjective surfaces". *MIT AI Memo* 666, 1981.

Brady, J. M., Horn, B. K. P. "Rotationally symmetric operators for surface interpolation". *Computer Graphics & Image Proc.*, 1983.

Brady, J. M., Yuille, Alan. "An extremum principle for shape from contour". *MIT A.I. Memo* 711, April 1983.

Brooks, R. A. "Symbolic reasoning among 3-D models and 2-D images". *Art. Intell.*, **17**, 285-348, 1981.

Canny, J. F. "A variational approach to edge detection", submitted to AAAI Conf., Washington, D. C., Sept., 1983.

Davis, L. S., Janos, L. & Dunn, S. "Efficient recovery of shape from texture". Computer Vision Lab., U. Maryland, *TR-1133*, 1982.

Horn, B.K.P. Woodham, R.J. and Silver W.M. "Determining shape and reflectance using multiple images," MIT AI Memo 490, 1978

Kanada, T. "Recovery of three-dimensional shape of an object from a single view". *Art. Intell.*, **17**, 409-460, 1981.

Kass, M. "A computational framework for the visual correspondence problem". This volume.

Marr, D. Vision, W. H. Freeman: San Francisco, 1982.

Marr, D. "Early Processing of visual information." *Phil. Trans. R. Soc. Lond. B*, **275**, 483 522, 1976.

Marr, D., Hildreth, E. "Theory of edge detection". *Proc. R. Soc. Lond. B*, **207**, 187-217, 1980.

Marr, D., Poggio, T. "A theory of human stereo vision". *Proc. R. Soc. Lond. B*, **204**, 301-328, 1979.

Marr, D., Ullman, S. "Directional selectivity and its use in early visual processing". *Proc. R. Soc. Lond. B*, **211**, 151-180, 1981.

Mayhew, J., Frisby, J. P. "Psychophysical and Computational studies toward a theory of human stereopsis". *Art. Intell.*, **17**, 349-387, 1981.

Nevatia, R., Binford, T. O. "Description and recognition of curved objects". *Art. Intell.*, 8, 77-98, 1977.

Nishihara, H. K. "Hidden information in early visual information processing". Society of Photo-optical instrumentation engineers technical symposium, 360, San Diego, 1982.

Nishihara, H. K. "Recognition of shape in visible-surfaces." In: **Physical & Biological Processing of Images**. Braddick and Sleigh, eds., Springer-Verlag, 1983.

Nishihara, H. K., Larson, N. G. "Towards a real-time implementation of the Marr-Poggio stereo matcher." *Proc. DARPA Image Understanding Workshop*, Baumann, ed., Science Applications, Inc., April, 1981.

Nishihara, H. K., Poggio, T. "Hidden cues in random line stereograms". *Nature*, **300**, 347-349, 1982.

Poggio, T. "Trigger features or Fourier analysis in early vision: a new point of view". In: *Recognition of Pattern and Form*, Albrecht, ed., Lecture notes in Biomathematics, 44, Springer-Verlag, 1982.

Poggio, T. "Visual Algorithms " in **Physical and Biological Processing of Images**, O.J. Braddick and A.C. Sleigh, Springer-Verlag, 1983.

Grimson, W. E. L. "A Computer implementation of a theory of human stereo vision". *Phil. Trans. Roy. Soc. Lond. B*, **292**, 217-253, 1981a.

Grimson, W. E. L. **From images to surfaces: a computational study of the human early visual system**, MIT Press, Cambridge, 1981b.

Grimson, W. E. L. "The implicit constraints of the primal sketch". *MIT A.I. Memo* 663, 1982a.

Grimson, W.E.L. "Binocular shading and visual surface reconstruction," *MIT AI Memo* 697, 1982b.

Grimson, W.E.L. "Aspects of a Computational Theory of Human Stereo Vision," Proceedings: Image Understanding Workshop, College Park, Maryland, 1980.

Grimson, W.E.L. and Marr, D. "A Computer Implementation of a Theory of Human Stereo Vision," Proceedings: Image Understanding Workshop, Palo Alto, California, 1979.

Haralick, R. M. "Zero-crossing of second directional derivative operator". *SPIE Proc. on Robot Vision*, Arlington, Virginia. 1982.

Hildreth, E. C. "The integration of motion information along contours". *Proc. Workshop on Computer Vision: Representation & Control*, Ringe, N.H., August 1982, 83-91.

Hildreth, E. C. "Computing the velocity field along contours". *Proc. ACM Interdisp. Workshop on Motion: Represent. & Percept.*, Toronto, Canada, 26-32, 1983.

Hildreth, E. C. "The detection of intensity changes by computer and biological vision systems". *Computer Graphics, Vision & Image Proc.*, in press.

Hildreth, E. C., Ullman, S. "The measurement of visual motion". *MIT A. I. Memo* 699, 1982.

Hollerbach, J. M. "Hierarchical shape description of objects by selection and modification of prototypes". *MIT A.I. Tech. Report.* 346, 1975.

Horn, B. K. P. "Understanding image intensities". *Art. Intell.*, 8, 201-231, 1977.

Horn, B. K. P., Schunck, B. G. "Determining optical flow". *Art. Intell.*, **17**, 1981.

Horn, B.K.P. and Sjoberg, R.W. "Calculating the reflectance map," Appl. Opt. 18, 11, (1979), 1770-1779.

THE SRI IMAGE UNDERSTANDING RESEARCH PROGRAM

M.A. Fischler (Principal Investigator)

SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

ABSTRACT

The ~~Our~~ principal objective in this research program is to obtain solutions to fundamental prohlems in computer vision; particularly those problems that are relevant to the development of an automated capability for interpreting aerial imagery and the production of cartographic products.

The authors ~~Our~~ plan is to advance the state of the art in selected core areas such as stereo compilation, feature extraction, linear delineation, and image matching; also, to develop an "expert system" control structure which will allow a human operator to communicate with the computer at a problem oriented level, and guide the behavior of the low level interpretation algorithms doing detailed image analysis.

Finally, it is planned ~~we plan~~ to use the DARPA/DMA Testbed as a mechanism for transporting both our own and IU community advances, in image interpretation and scene analysis, to DMA, ETL, and other members of the user community.

I INTRODUCTION

A major focus of our current work is the construction of an Expert System for Stereo Compilation and Feature Extraction. Our intent in this effort is to develop a system that provides a framework for allowing higher level knowledge to guide the detailed interpretation of imaged data by autonomous scene analysis techniques. Such a system would allow symbolic knowledge, provided by higher level knowledge sources, to automatically control the selection of appropriate algorithms, adjust their parameters, and apply them in the relevant portions of the image.

Recognizing the difficulty of completely automating the interpretation process, the expert system will be structured so that a human operator can provide the required high level information when there are no reliable techniques for automatically extracting this information from the available imagery. As new research results become available, the level of human interaction can be progressively reduced.

The expert system we are building can thus be viewed as an intelligent user-level interface for guiding semiautomated image processing activities. Such a system is envisioned as a rule-based system with a library of processes and activities, which can be invoked to carry out specific goals in the domain of cartographic analysis and stereo reconstruction. The system would depend on the human user for those types of information not easily extracted from the given imagery, and allow the computer system to take over in those areas where the utility of automated analysis has been clearly demonstrated.

Development of the expert system control structure is a research task still in an early stage of accomplishment. The remainder of this report will describe progress in research supporting the development of potential scene analysis components of the system, as well as other Image Understanding research of a more basic nature. We also briefly describe the status of the DARPA/DMA Testbed effort now approaching completion.

II RESEARCH PLANS AND PROGRESS

A. Development of Methods for Modeling and Using Physical Constraints in Image Interpretation.

Our goal in this work is to develop methods that will first allow us to produce a sketch of the physical nature of a scene and the illumination and imaging conditions, and next permit us to use this physical sketch to guide and constrain the more detailed descriptive processes -- such as precise stereo mapping.

Our approach is to develop models of the relationship between physical objects in the scene and the intensity patterns they produce in an image (e.g., models that allow us to classify intensity edges in an image as either shadow, or occlusion, or surface intersection, or material boundaries in the scene); models of the geometric constraints induced by the projective imaging process (e.g., models that allow us to determine the location and orientation of the camera that acquired the image, location of the vanishing points induced by the interaction between scene and camera, location of a ground plane, etc.); and models of the illumination and intensity transformations caused by the atmosphere, light reflecting from scene surfaces, and the film and digitization processes that result in the computer representation of the image.

These models, when instantiated for a given scene, provide us with the desired "physical" sketch. We are assembling a "constraint-based

Poggio, T., Nishihara, H. K., Nielsen, K. R. K. "Zero-crossings and spatiotemporal interpolation in vision: aliasing and electrical coupling between sensors". *MIT A. I. Memo* 675, 1982.

Stevens, K. A. "The visual interpetation of surface contours". *Art. Intell.*, 17, 47-75, 1981.

Terzopoulos, D. "Multi-level reconstruction of visual surfaces". *MIT A. I. Memo* 671, 1982.

Terzopoulos, D. "The role of constraints and discontinuities in visible-surface reconstruction", *Proc. Int. Jt. Conf. Art. Intell.*, Karlsruhe, 1983.

Terzopoulos, D. "Multi-level reconstruction of visual surfaces". *MIT A. I. Memo* 671, 1983b. To appear in **Multiresolution Image Processing and Analysis**, A Rosenfeld, ed.

Witkin, A. P. "Shape from contour". Ph.D. thesis, MIT. Also *MIT A.I. Tech. Report* 589, 1980.

Woodham, J. R. "Analyzing images of curves surfaces". *Art. Intell.*, 17, 117-141, 1981.

Ullman, S., Hildreth, E. C. "The measurement of visual motion". In: **Physical and Biological Processing of Images**, Braddick and Sleigh, eds., Springer-Verlag, Berlin, 1983.

Yuille, A. "Zero crossings on lines of curvature". Submitted to AAAI Conf., Washington, D. C., Sept., 1983.

stereo system" that can use this physical sketch to resolve the ambiguities that defeat conventional approaches to stereo modeling of scenes (e.g., urban scenes or scenes of cultural sites) for which the images are widely separated in either space or time, or for which there are large featureless areas, or a significant number of occlusions.

Recent publications of our work in this area are cited in the references [1-4, 9-12].

B. Stereo Compilation: Image Matching and Interpolation

We are implementing a complete state-of-the-art stereo system that produces dense range images from given pairs of intensity images. We plan to use this system both as a framework for our stereo research, and as the base component of our planned expert system.

There are five components of this stereo system: a rectifier, a sparse matcher, a dense matcher, an interpolator, and a projective display module. The rectifier estimates the parameters and distortions associated with the imaging process, the photographic process, and the digitization. These parameters are used to map digitized image coordinates onto an ideal image plane. The sparse matcher performs two-dimensional searches to find several matching points in the two images, which it uses to compute a relative camera model. The dense matcher tries to match as many points as possible in the two images. It uses the relative camera model to constrain the searches to one dimension, along epipolar lines. The interpolator computes a grid of range values by interpolating between the matches found by the dense matcher. The projective display module allows interactive examination of the computed 3-D model by generating 2-D projective views of the model from arbitrarily selected locations in space. Initial versions of all components of the system have been implemented.

Present research in this task is focused primarily on the image correspondence (matching) and interpolation problems. With respect to image matching, the following major issues are being addressed:

- What is a correct match?
- How does one measure the performance of a matcher?
- What causes existing matching techniques to fail?
- How can one improve the performance of matching techniques?

Since there are no reliable analysis techniques for evaluating the performance of matching algorithms when applied to real world images, we must evaluate them by extensive testing. To expedite such testing, a database of images and ideal match data (ground truth) is being assembled. For example, we have acquired data from the ETL Phoenix test site that were produced specifically for testing matching techniques. Every point in the database we are constructing contains annotations that indicate the categories of matching problems for that point, and other information that might be useful to evaluate the performance or guide the application of matching techniques.

We are currently investigating a hypothesize - verify approach to local matching. Potential matches are verified by examining the image for compliance with the assumptions of the matching operator's model. For example, area correlation matching operators assume that correctly registered image patches will differ only by Gaussian noise. A simple verification technique is to examine the statistics of the point-by-point difference between the hypothesized alignment of the patches for conformance with that model. Image anomalies, such as moving objects or occluding contours, will typically produce a difference image that has a highly structured geometry, indicating the shape and location of the anomaly. Such anomalous areas can be removed from the region over which the correlation is computed, and the process iterates until either an acceptable match criterion is satisfied, or too many points are removed from the region.

In many cases (e.g., occlusion and featureless areas) local matching techniques are not capable of producing the required correspondences over regions of significant extent. We intend to use the information provided by the "physical sketch" (see previous section) to detect such situations, and to select alternative means for obtaining the required depth information.

As indicated above, when a stereo pair of images are matched, we generally can do no better than to compute a sparse depth map of the imaged scene. However, for many tasks a sparse depth map is inadequate. We want a complete model that accurately portrays the scene's surfaces. To achieve this goal, we must be able to obtain the missing surface shape information from the shading of the images of the stereo pair.

To understand the relationship between image shading and surface shape, we built a differential model [10,11] that relates shape and shading but, unfortunately, does not provide a complete basis for a shape recovery algorithm [12]. However, the information available in image shading does allow the building of a surface interpolation algorithm that finds a surface that is consistent with the image shading. We are proceeding with such a development.

As image shading alone does not provide sufficient information to find surface orientation, further shape information sources in the image are needed. We are evaluating additional scene attributes that encode shape information in their image, and the models necessary to recover the corresponding shape information.

C. Feature Extraction: Scene Description, Partitioning, and Labeling

Our current research in this area addresses two related problems: (1) representing natural shapes such as mountains, vegetation, and clouds, and (2) computing such descriptions from image data. The first step towards solving these problems is to obtain a model of natural surface shapes.

A model of natural surfaces is extremely important because we face problems that seem impossible to address with standard descriptive computer vision techniques. How, for instance, should we describe the shape of leaves on a tree? Or grass? Or clouds? When we attempt to describe such common, natural shapes using standard shape-primitive representations, the result is an unrealistically complicated model of something that, viewed introspectively, seems very simple. Furthermore, how can we extract 3-D information from the image of a textured surface when we have no models that describe natural surfaces and how they evidence themselves in the image? The lack of such a 3-D model has restricted image texture descriptions to being ad hoc statistical measures of the image intensity surface.

Fractal functions, a novel class of naturally-arising functions, are a good choice for modeling natural surfaces because many basic physical processes (e.g., erosion and aggregation) produce a fractal surface shape, and because fractals are widely used as a graphics tool for generating natural-looking shapes. Additionally, we have recently conducted a survey of natural imagery and found that a fractal model of imaged 3-D surfaces furnishes an accurate description of both textured and shaded image regions, thus providing validation of this physics-derived model for both image texture and shading.

Encouraging progress relevant to computing 3-D information from imaged data has already been achieved by use of the fractal model. We have derived a test to determine whether or not the fractal model is valid for particular image data, developed an empirical method for computing surface roughness from image data, and made substantial progress in the areas of shape-from-texture and texture segmentation. Characterization of image texture by means of a fractal surface model has also shed considerable light on the physical basis for several of the texture partitioning techniques currently in use, and made it possible to describe image texture in a manner that is stable over transformations of scale and linear transforms of intensity.

The computation of a 3-D fractal-based representation from actual image data has been demonstrated. This work has shown the potential of a fractal-based representation for efficiently computing good 3-D representations for a variety of natural shapes, including such seemingly difficult cases as mountains, vegetation, and clouds.

This research is expected to contribute to the development of (1) a computational theory of vision applicable to natural surface shapes, (2) compact representations of shape useful for natural surfaces, and (3) real-time regeneration and display of natural scenes. We also anticipate adding significantly to our understanding of the way humans perceive natural scenes.

Details of this work can be found in Pentland [8].

D. Linear Delineation and Partitioning

A basic problem in machine vision research is how to produce a line sketch that adequately captures the semantic information present in an image. (For example, maps are stylized line sketches that depict restricted types of scene information.) Before we can hope to attack the problem of semantic interpretation, we must solve some open problems concerned with direct perception of line-like structure in an image and with decomposing complex networks of line-like structures into their primitive (coherent) components. Both of these problems have important practical as well as theoretical implications.

For example, the roads, rivers, and rail-lines in aerial images have a line-like appearance. Methods for detecting such structures must be general enough to deal with the wide variety of shapes they can assume in an image as they traverse natural terrain.

Most approaches to object recognition depend on using the information encoded in the geometric shape of the contours of the objects. When objects occlude or touch one another, decomposition of the merged contours is a critical step in interpretation.

We have recently made significant progress in both the delineation and the partitioning problems. Our work in delineation [5] is based on the discovery of a new perceptual primitive that is highly effective in locating line-like (as opposed to edge-like) structure.

Our work on decomposing linear structures into coherent components [6] is based on the formulation of two general principles that appear to have applicability over a wide range of problems in machine perception. The first of these principles asserts that perceptual decisions must be stable under at least small perturbations of both the imaging conditions and the decision algorithm parameters. The second principle is the assertion that perception is an explanatory process: acceptable precepts must be associated with explanations that are both complete (i.e., they explain all the data) and believable (i.e., they are both concise and of limited complexity).

These new delineation and partitioning algorithms have produced excellent results in experimental tests on real data [5,6].

III STATUS OF THE DARPA/DMA IMAGE UNDERSTANDING TESTBED

The DARPA/DMA Image Understanding Testbed established at SRI as part of the DARPA Image Understanding research program constitutes a coherent body of software running in a standard hardware environment. Demonstrations of the features and capabilities of all IU community contributed software are available; detailed evaluations have been carried out for selected modules (e.g., see the paper by K. Laws [7] in these proceedings). In this capacity, the Testbed

INTELLIGENT VISION SYSTEMS

Thomas O. Binford
for the staff of

Artificial Intelligence Laboratory, Computer Science Department
Stanford University, Stanford, California 94305

Abstract

Research on intelligent systems for image understanding focusses on a successor to the ACRONYM system, and its application in a rule-based stereo mapping and interpretation system. Some elements of a rule-based stereo system have been implemented. A new modeling system is under construction, and a new graphics system for display of generalized cylinders has achieved initial results. Research has continued on segmentation/aggregation in the figure-ground problem for grouping candidate objects. Implementation experiments are underway for an array of vision processors. Fundamental mathematical results have been obtained on matching processes. Inference rules for interpreting surfaces from images were demonstrated formally in a mathematical logic programming system. Results have been obtained in specializing certain vision programs by automatic methods to produce efficient programs.

The objectives of this research are to develop algorithms for high performance image understanding modules, to implement an intelligent vision system, and to demonstrate its application in photointerpretation and cartography. The ACRONYM system was developed as the first intelligent system. Research has shifted to its SUCCESSOR.

A rule-based stereo mapping system is under construction. Various members of the group have built elements for a demonstration described in [Baker 83]. This work was supported in part by RADC. These include: an evaluation of Pentland's shape from shading program; an extended version of Baker's program which includes edges from [Marimont 82]; a stereo registration and rectification program by Meller; generic building models and typical building examples by Gray; stereo matching of orthogonal trihedral vertices by Malik and Binford; monocular and stereo inference rules by Malik and Binford; example rules for a rule-based stereo system.

Miller and Lowry have continued progress toward building a small array of image processors [Lowry 82]. Other work has begun in the architecture of algorithms for image understanding.

Cowan has begun implementation of a new modeling system for SUCCESSOR. Rublee and Selker have investigated the user interface for an intelligent geometric editor. Chelberg has investigated the constraint system and rule base of ACRONYM, using a large set of aircraft models. Minor problems were identified and fixed. He is investigating more powerful mechanisms for the constraint system. Lowry has done initial work in problem formulation for a class of computational geometry problems. [Scott 83] has implemented a general system for calculating the terminators (visible boundaries of curved surfaces) for a broad class of generalized cylinder models. The algorithm, capable of parallel implementation, calculates the perspective image of a Generalised Cylinder, from arbitrary viewpoint, with hidden surface removal. It applies to a wide class of cylinders. The time taken will be proportional to the total length of the contours, independent of the number of edges. The algorithm solves for one closed-loop contour-generator at a time, testing its contour (in the image plane) for intersection with visible segements of previous contours.

[Lowe 83] have extended the analysis of the figure/ground problem, which we formulate as the discovery of non-random structure in images, whether interpreted as surfaces in three space, or as patterns and texture in the plane. Uniform, non-random structure has an interpretation of common phyiscal origin. Marimont has worked at finding edges in intensity surfaces. As a subproblem of segmenting intensity surfaces, he has investigated segmenting curves. Results have been obtained for the problem of determining a smooth curve through two samples, each with point, tangent vector, and curvature.

[Blicher 83] has developed some fundamental mathematical theory underlying vision. He defines a mathematical structure which can be used as a framework for studying many vision problems. Drawing on differential topology, he uses the framework to prove a theorem regarding the stereo matching problem. The main result is that without constraints on imaging geometry, matching of typical pictures requires at least 2 color dimensions for uniqueness. He also presents some theory about the topology of iso-brightness contour lines, which is useful in understanding the behavior of systems which track some value, e.g. zero-crossings. The paper provides vision researchers with a view of some of the powerful results of modern differential topology; the methods used are applicable to stereo, motion stereo, optic flow, and matching.

[Ketonen 83] is investigating ways of formally expressing facts about images. In particular, he can show that some of the coincidence assumptions stated in [Binford 81] can actually be proved in a suitable formal framework.

[Goad 83] describes the automatic generation of special purpose vision programs. The starting point for the automatic construction process is a description of a particular 3D object. The result is a special purpose program for recognizing and locating that object in images, without restriction on the orientation of the object in space. Thus each object description is analyzed in advance, and then "compiled" into an efficient program for detecting that object in images. The method has been implemented and tested on a variety of images with good results. Some of the tests involved images in which the target objects appear in a jumbled pile. The current implementation is not fully optimized for speed. However, evidence is given that image analysis times on the order of a second or less can be obtained for typical industrial recognition tasks. (This time estimate excludes edge finding).

Perceptual Organization

We have a practical objective which is to implement more general interpretation in ACRONYM. A short term goal is an improved ribbon finder, coupled with a mechanism for making canonical clusters of ribbons. ACRONYM matches predicted ribbons or is now established as a technology transfer tool that can be utilized by appropriate agencies to evaluate the applicability of the contributed scene analysis techniques.

Documentation of the Testbed is entering its final phase. Final drafts of the User's manual, the Programmer's manual, and the System Manager's manual are available and will soon pass through the required editing and approval procedures. Drafts of the evaluation reports for the Ghough and Phoenix programs are also complete. We are currently completing both the evaluation report for the Relaxation package and the user-level documentation of those contributions for which no detailed evaluation is planned. More extensive studies of the various approaches to stereo compilation now available on the Testbed will be integrated into the ongoing research effort on the stereo problem.

The Testbed is now sufficiently well-defined that exact copies of the entire system can be configured, if desired. SRI, under a separate contract, is just completing the installation of a Testbed copy (hardware and software) at the US Army Engineer Topographic Laboratories (ETL) at Fort Belvoir. A Lisp Machine will be added to the ETL configuration later in the year. SRI will also be supplying Lisp Machines and Lisp Machine software to the DMAHTC and DMAAC branches of the Defense Mapping Agency. SRI has been closely involved in efforts to ensure that the upgrade of the DMA AFES/RWPF facilities to the VAX-11/780 CPU can incorporate the Image Understanding Testbed capabilities, as well as supporting the Lisp Machines.

The Testbed software system and its utilities are being prepared for export to university researchers in the IU program as well as to other U.S. Government agencies interested in establishing Testbed copies. SRI has developed a simple license agreement to help protect Testbed contributors and restrict use of the software to appropriate academic and government research environments.

ACKNOWLEDGEMENT

The following researchers have contributed to the work described in this report: S. Barnard, R.C. Bolles, M.A. Fischler, M.J. Hannah, A.J. Hanson, D.L. Kashtan, K. Laws, A. Pentland, L.H. Quam, G.B. Smith, and H.C. Wolf.

REFERENCES

1. S. Barnard and A. Pentland, "Three-Dimensional Shape from Line Drawings," these proceedings and IJCAI-83.

2. S. Barnard, "Methods for Interpreting Perspective Images," (in press) AI Journal, 1983.

3. S. Barnard and M.A. Fischler, "Computational Stereo," ACM Computing Surveys, Vol. 14 (4), December 1982.

4. M.A. Fischler, et.al., "Modeling and Using Physical Constraints in Scene Analysis," AAAI-82.

5. M.A. Fischler and H.C. Wolf "Linear Delineation," IEEE CVPR-83.

6. M.A. Fischler and R.C. Bolles "Perceptual Organization and Curve Partitioning," these proceedings and IEEE CVPR-83.

7. K. Laws, "On the Evaluation of Scene Analysis Algorithms," these proceedings.

8. A. Pentland, "Fractal Based Description of Natural Scenes," these proceedings and IEEE CVPR-83.

9. A. Pentland, "Depth of Scene from Depth of Field," Proceedings of the Image Understanding Workshop, September 1982.

10. A. Pentland, "Local Analysis of the Image: Limitations and Uses of Shading," Proceedings of the (IEEE) Workshop on Computer Vision: Representation and Control, Rindge, New Hampshire, August 1982.

11. G.B. Smith, "The Recovery of Surface Orientation from Image Irradiance," Proceedings of the Image Understanding Workshop, September 1982.

12. G.B. Smith, "The Relationship Between Image Irradiance and Surface Orientation," these proceedings and IEEE CVPR-83.

ellipses with observed ribbons or ellipses; then it tests whether clusters of image elements satisfy object constraints in three space. Typically, matching single ribbons is weak, while matching all pairs, triples, and n-tuples of ribbons is combinatorially unattractive. Limiting the combinatorics leads to introducing proximity grouping, then to a thorough investigation of grouping mechanisms from first principles.

We have studied fundamental properties of perceptual organization. The bottom-up process of grouping related image features plays an important role in 3-D inference, model-based recognition, and matching processes such as stereo correspondance. We measure the significance of image relations as inversely proportional to the probability that they would have arisen by accident from the surrounding distribution of features. This is a general measure that requires no prior knowledge of the scene, and can therefore be applied uniformly at the earliest stages of the image interpretation process.

Because the image relations are likely to have arisen from properties of the scene rather than through an accident of image formation, they provide a reliable basis for matching against models. ACRONYM currently relies on ribbons and elipses as its perceptual description of an image, but this set could be expanded to include all reliably detectable relations. Typically relations which are non-accidental will be quasi-invariant with respect to viewpoint. This means that these relations can be used for stereo correspondance matching, at a higher and more robust level than simple edge points.

We have also studied the complexity of the process of forming image relations. It would be combinatorially expensive to examine all possible relations between image features. Therefore, we have used diameter-limited grouping processes applied at multiple scales and overlapping locations. At any scale, the number of alternatives for forming relations must be low, or none will be attempted. In this way, computation is limited by complexity rather than by prior limits on scale or density. Some of this work is the basis for estimates concerning architecture of intermediate-level vision.

We have currently implemented a curve description program which looks for non-accidental linear or curvilinear structure in edge data. This program is able to detect significant structure occuring at multiple scales in the same edge. It requires no prior knowledge of the noise properties in an edge, but uses the given data to estimate the scales at which the curve exhibits the most significant structure.

Stereo Vision

Baker has modified the stereo system to include edges from [Marimont 82]. The system now uses improved edge operators and includes edge extent in seeking optimal correspondence. The system now deals with stereo pairs in which epipolar lines are not coincident with the camera raster. To bring this about, Meller made a program to determine epipolar lines from the camera transform data of [Gennery 77]. To perform the interpolation of surfaces based on intensity interpolation, Meller's program was made to produce images rectified to epipolar geometry.

A system was developed for input of hand-segmented images as a basis for developing higher level inference and correspondence functions independent of the development of segmentation algorithms.

Orthogonal trihedral vertices (OTVs) are an important structural element in buildings. OTVs were analyzed in part by [Liebes 81]. Malik and Binford provided an analysis for general orientation in perspective. The analysis was implemented as an inequality to test candidate OTVs; all OTVs and some non-OTVs are accepted by the inequality. An algorithm determines the angle in space from a single image. Clearly, corresponding views of an OTV must imply the same orientation in space.

Malik and Binford have determined new monocular inference rules which have applicability to stereo, including a generalized support interpretation. They have produced a stereo inference rule which imposes a sign reversal constraint on pairs of vectors. If two images of a pair of vectors are to correspond, the z-component of their cross product (determined entirely by image quantities) must not change sign.

Segmentation and Representation of Curves

We have developed an algorithm to compute a segmentation and representation of digital curves, applicable to edges extracted from images, intended to facilitate higher-level analysis of curves. A number of psychological and mathematical considerations have led us to segment curves at extrema and zeroes of estimated curvature. Psychological data suggest that humans segment curves at extrema, and that humans are insensitive to derivatives of order higher than two (curvature is closely related to the second derivative). Further, zeroes and extrema of curvature have mathematical properties of invariance under certain geometric transformations which enable reliable estimation of curvature characteristics independent of the curve's position and orientation. A related conjecture currently being investigated is whether suitably chosen invariants of space curves map stably under perspective projection into extrema and zeroes of curvature of the image plane curve.

We estimate curvature at all scales, and a pyramid of curvature estimates is constructed suitable for detection and representation of linear and hierarchical relationships among the estimates. We use this pyramid to evaluate robustly the significance of of curvature changes at one scale in the context of others; we thereby eliminate the need for extensive prior knowledge of sensor noise, for instance. Estimates of significant curvature changes are retained at all scales, so tasks needing only rough estimates are not computationally overburdened by unnecessary detail, while those able to use high accuracy effectively achieve optimal performance.

Splines for Vision

We have completed a preliminary implementation of a new type of spline based on intrinsic, geometric properties of curves. We argued above that digital curves should be segmented at extrema and zeroes of curvature. This new spline takes as input two points, two tangents, and two curvatures, and returns a curve which is: in agreement with the input data at the two points; continuous; continuous in tangent; continuous in curvature, with curvature varying monotonically along the curve. Curvatures at the endpoints cannot be of different signs, although one can be zero and one nonzero. If our curvature estimates are consistent with the assumption that curvature is continuous, this restriction poses no problem, since placing knots at all zeroes and extrema of curvature implies that no two adjacent knots can have curvatures of opposite sign (if they did, there would be a zero of curvature between them, and therefore a knot).

Curvature must change monotonically between knots to avoid introducing spurious curvature extrema, i.e. extrema not present in the curve underlying our curvature estimates. If the curvatures at the two points to be splined are k_1 and k_2, with k_1 less than k_2, then the statement that curvature increases monotonically betwee k_1 and k_2 is mathematically equivalent to the statement that there exist no curvature extrema between k_1 and k_2 (assuming curvature is continuous). Since the perceptual and mathematical importance of curvature extrema dictated the placement of knots at them, it is crucial that the spline introduce no curvature extrema not present in the data. In recognition of the importance of this characteristic, we refer to these splines as monotone curvature splines. The current implementation relies on the relationship between evolutes and involutes, a construct from classical differential geometry. The spline itself is the involute, determined by a trivial calculation from the evolute; finding the evolute is the computationally hard part. The evolute is not determined uniquely from the input data. We have chosen to use four circular arcs, primarly for the sake of computational efficiency; the resulting evolute is continuous, and continuous in tangent, but not in curvature. It is a fortuitous aspect of the relationship between evolutes and involutes that the involute's curvature is continous. An iterative procedure is used to find the evolute; it converges in test cases satisfactorily and rapidly, although more testing needs to be done.

Prediction of Generalized Cylinders

This is the first stage of a system for manipulating generalized cylinder models. It includes a generalization of the formulation of [Shafer 82] for the prediction of the terminator for generalized cylinders. The algorithm can be divided into two parts. First (A), solution for the visible parts of the contour-generators, and secondly (B), region growing to get visible surfaces. The first part is the principal one. It has two subparts, which are repeated, and together find one contour-generator. Each contour-generator is a closed loop, intersecting no others, which divides the surface into forward (visible if unoccluded), and backward facing (invisible) areas. The square root of the size of each visible area, is a measure of the length scale over which things are happening in that region of the GC.

The first subpart (A1), steps over the GC with step length proportional to the square root of the area of the region it is contained by, until either the whole surface has been covered, in which case the algorithm stops; or a step containing a new contour-generator is found. In this case, the step is then bisected down to an exact solution. A test to see whether each step jumps a new contour-generator can be made since, whenever the direction (forward or back), that a surface point is facing, differs from the direction predicted by the regions of the existing contour-generators, then there must be an undiscovered contour-generator passing nearby. This means that if one stepping point has the same predicted, and actual surface direction, and the next does not, then a new contour generator passes through the intervening step. This interval is reduced, using bisection, with the condition that one end of the interval must have the same predicted, and real surface directons, while the other end must not.

The solution is handed over to the second subpart (A2), which propagates it around the whole contour-generator, back to its start, making a list of the solutions as it goes, and noting the ones where the contour-generator becomes visible or occluded. It works by stepping along the contour-generator tangent 2, to get a guess for the next solution point, which is Newton-Raphson iterated to a sufficient accuracy. If the Newton-Raphson does not converge, several points around a small circle are tested to find an interval to bisect down to the next solution. The step length is taken proportional to curvature of the contour, to get uniform interpolation accuracy between the known contour points3,1. Each step is projected to the image, and checked for intersection with those previously projected steps, which have not been shown to be hidden. When an intersection is found, the exact positions of the occluding and occluded contour-generator points are calculated6. Finally the whole contour is checked against possible surrounding contours.

To convert to a form implementable in parallel; step A1 is done independently, at different places and then A2 is used to form contour- generator segments, which can be simply joined up into the complete contour lists. Either way, each list of contour points is now followed down, keeping count of the marked occluded points, to produce lists of just the visible ones.

Automatic Generation of special purpose vision programs

Chris Goad's work concerns the automatic generation of special purpose vision programs.

In many practical applications of automated vision, the vision task takes the form of recognizing and locating a particular three dimensional object in a digitized image. The exact shape of the object to be perceived is known in advance; the purpose of the act of perception is only to determine its position and orientation relative to the viewer. This is model based vision in its strict form. Most industrial applications of vision have this property, and also the property that the same object (or, more precisely, objects of the same shape), must be located in many images.

Goad's work concerns a scheme for exploiting this kind of situation which involves automatically constructing special purpose vision programs. The starting point for the automatic construction process is a description of a particular 3D object. The result is a special purpose program for recognizing and locating that object in images, without restriction on the orientation of the object in space. Since this special purpose program has a comparatively limited task to perform, it can be much faster than any general purpose vision program would be. Thus each object model is analyzed in advance, and then "compiled" into an efficient program for detecting that object in images. The method has been implemented and tested on a variety of images with good results. Some of the tests involved images in which the target objects appear in a jumbled pile. The current implementation is not fully optimized for speed. However, evidence is given that image analysis times on the order of a second or less can be obtained for typical industrial recognition tasks. (This time estimate excludes edge finding).

Mathematical Analysis

from the camera transform data of [Gennery 77]. To perform the interpolation models. It includes a generalization of the formulation of [Shafer 82] [Ketonen 83] has implemented a formal representation of geometry in the EKL system. He has demonstrated that some of the coincidence assumptions stated in [Binford 81] can actually be proved in a suitable formal framework.

It follows from his analysis that many of the "impossible" pictures of Huffman in [2] can be detected by simpler and more general means than the ones used by Huffman, Clowes or Waltz. Given that these methods are simpler (even if not complete), they may be closer to the process actually used by the human visual system.

One should not expect formalisations of theories to have tangible connections with succesful implementations of algorithms; Artificial Intelligence programs need not be based on the paradigm of theorem proving. However, the clarification of the formal concepts underlying these systems can be of great importance in terms of program architecture and further development.

In Blicher's work, a unifying abstract mathematical structure is presented for a number of vision problems, notably stereo, motion stereo, optic flow, and matching. The structure is summarized in Fig. (*') of Blicher's paper; he defines the various

Dup

PRINCIPAL INVESTIGATORS' REPORT ON CONTRACT DAAG53-76-C-0138, "UNDERSTANDING OBJECTS, FEATURES, AND BACKGROUNDS", AND CONTRACT DAAK70-83-K-0018, "AUTONOMOUS VEHICLE NAVIGATION"

Azriel Rosenfeld
Larry S. Davis

Center for Automation Research
University of Maryland
College Park, MD 20742

ABSTRACT

This report summarizes the work done during the final two years of Contract DAAG53-76-C-0138, "Understanding Objects, Features, and Backgrounds." It also outlines plans for work to be conducted during the coming three years on Contract DAAK70-83-K-0018, "Autonomous Vehicle Navigation."

1. UNDERSTANDING OBJECTS, FEATURES, AND BACKGROUNDS

1.1. Introduction

In June 1976 the U.S. Army Night Vision and Electro-Optics Laboratory awarded Contract DAAG-53-76-C-0138 to the University of Maryland for research on "Algorithms and Hardware Technology for Image Recognition." Funding for this contract was derived primarily from the Defense Advanced Research Projects Agency under DARPA Order 3206. During the following 21-month period, the University developed and tested advanced algorithms for detection of tactical targets on Forward-Looking InfraRed (FLIR) imagery. Concurrently, on a subcontract, the Westinghouse Defense Systems Division designed charge-coupled device (CCD) layouts for implementing many of these algorithms in hardware, and also fabricated a CCD chip that implemented one basic algorithm, histogramming/sorting. The results of the work done during the first 21 months of the contract are documented in detail in a Final Report dated March 1978 [A1].

In April 1978 the contract was extended for a two-year period, under the new title "Image Understanding Using Overlays." During this phase of the project, numerous algorithms were developed and tested for object detection and extraction from images, as well as for image and region representation. On a subcontract, Westinghouse investigated the implementation of some of these algorithms in general- or special-purpose digital hardware. Westinghouse also conducted tests of one class of algorithms known as "relaxation" techniques. The results of the work done during this period are documented in a series of technical and semiannual reports, are are summarized in a Final Report dated May 1980 [A2].

In May 1980 the contract was extended for a final two-year period (later extended, at no additional cost, through December 1982), under the title "Understanding Objects, Features, and Backgrounds." During this phase of the project, further studies were conducted, in collaboration with Westinghouse, on object segmentation and recognition, feature extraction and background analysis, multi-resolution image processing techniques, and analysis of time varying imagery. This work was documented in a series of project status reports [B1-3] and Technical Reports [C1-32], and is summarized in this Final Report.

Principal Investigators on this project at the University of Maryland were Profs. Azriel Rosenfeld and Larry S. Davis, and at Westinghouse, Dr. Glenn E. Tisdale and Mr. Bruce J. Schachter. The project monitor at NVEOL is Dr. George R. Jones.

1.2. Object segmentation and recognition

a) Comparative segmentation study

A comparative study of object extraction techniques applicable to FLIR imagery was conducted jointly by Maryland and Westinghouse, using a database of 52 images collected by Westinghouse from Army, Navy, and Air Force sources. Techniques tested by Maryland included two variations of a relaxation method as well as new methods based on multiresolution image representations, known as "pyramids." One of the pyramid-based methods outperformed all the other techniques tested. The results of the Maryland study are documented in detail in a technical report [C19], while Westinghouse's study is documented in a Westinghouse report.

b) New segmentation techniques

As a supplement to the main segmentation study, several new segmentation techniques were developed under the project. Two methods developed on earlier projects were extended from single-band to multiband imagery. One of these improves the detectability of clusters in a histogram or scatterplot by suppressing pixels that lie on edges [C1]. The other, known as "Superspike," converts the peaks in a histogram or scatterplot into sharp spaces and mappings present in performing mathing, and their relationships and properties. This is done in a fairly abstract way, so as to be applicable to many different types of vision problem. For example, the same formalism describes perspective as well as orthogonal projection, unusual camera geometries, and projection onto a plane or a sphere, etc. Blicher believes that this type of language can eventually be converted into a computer language for describing a computational environment for vision.

Ideas from modern differential topology are presented and applied to the general matching problem, a common approach to stereo matching, defined as follows. Given 2 picture functions $F_1, F_2 : M^2 \rightarrow R^n$, one finds regions $K_1, K_2 \subset M^2$ and a 1-1 *matching* function $g_\pi : K_1 \rightarrow K_2$ such that $F_1 = F_2 \circ g_\pi$. Blicher proves a "2-color theorem": that *generically* for monochrome pictures ($n = 1$) there is a large infinity of solutions, but for 2 or more colors ($n \geq 2$) the solution is unique. In the monochrome case, the solutions can be quite different, matching the same point to widely separated target points. Though the theorem literally deals with matching grey levels, it is equally valid for a derived function, such as the output of a lateral inhibition operator, a smoother, or an edge filter, although only areas lacking occlusion are considered.

"Generic" is a central concept in differential topology, which means "almost always" in a precise way, allowing one to exclude pathological or unlikely behaviors which cannot be encountered in practice, thus making many problems tractable. This can find application as well in inferring structure from images.

The proof of the theorem for the monochrome case is based on a very simple intuitive argument involving sliding isobrightness contours along themselves. Independently of proving the theorem, to flesh out the intuition, Blicher presents some facts about how such contour lines can look. Although no use is made of it in the paper, such information in itself is useful for matching, as topological structure is invariant for small perturbations, hence it is important to classify the possibilities into a small discrete set. Also, this theory is useful for understanding any real-valued function on a picture, for example the zero-crossings of an edge finder, or the values of some curvature parameter, say Gaussian curvature, or even some local Fourier coefficient, as one might use for a texture system.

—— equation-free sentence:

Ideas from modern differential topology are presented and applied to the general matching problem, a common approach to stereo matching, defined as follows.

> Given 2 picture functions, one finds 2 regions and a 1-1 *matching* function between them such that the matching function preserves grey level values.

References

[Baker 83] H.Baker; "Progress in stereo mapping"; Proc Image Understanding Workshop, April 1983.

[Binford 81] T.Binford, Inferring Surfaces from Images, AI Journal, (August 1981).

[Blicher 83] "Stereo Matching from the Topological viewpoint"; Proc Image Understanding Workshop, April 1983.

[Gennery 77] Gennery, D.B.; "A Stereo Vision System for An Autonomous Vehicle"; Proc 5th International Joint Conf on Artificial Intelligence, MIT, Boston, Aug 1977. Gennery, D.B.; "Stereo Camera Solver"; Stanford A I Lab Internal Note, 1976.

[Goad 83] C. Goad; "Special Purpose Automatic Programming for 3-D Model-Based Vision"

[Huffman 71] D.A.Huffman, Impossible Objects as Nonsense Sentences, Machine Intelligence 6, (1971), 295-324.

[Ketonen 83] "Deducing Facts about Scenes from Images" Proc Image Understanding Workshop, April 1983.

[Liebes 81] Liebes, Jr., S.; "Geometric Constraints for Interpreting Images of Common Structural Elements: Orthogonal Trihedral Vertices"; Proc Image Understanding Workshop, April 1981.

[Lowe 82] David Lowe and T.O.Binford; "Segmentation and Aggregation: Figure-Ground Phenomena"; Proc. IU Workshop Image Understanding September 1982.

[Lowe 83] David Lowe and T.O.Binford; "The Perceptual Organization of Visual Images: Segmentation as a basis for Recognition;" Proc. IU Workshop Image Understanding September 1983.

[Lowry 82] Michael Lowry and Allan Miller; "Analysis of low-level computer vision" algorithms for implementation on a VLSI processor array"; Proc. IU Workshop Image Understanding September 1982.

[Malik 82] J.M. Malik and T.O. Binford; "Representation of time and sequences of events"; Proc. IU Workshop Image Understanding September 1982.

[Marimont 82] David Marimont; "Segmentation in Acronym"; Proc. IU Workshop Image Understanding September 1982.

[Scott 83] R. Scott; An algorithm to display generalised cylinders. Proc. IU Workshop Image Understanding September 1983.

[Shafer 82] S.Shafer, T.Kanade, The Theory of Straight Generalised Cylinders.

spikes (thus making them trivially detectable) by a process of iterated local averaging in which the histogram is used as a guide in selecting those neighbors with which a given pixel should be averaged [C26]. A third method, "bimean clustering," identifies the two "best" subpopulations in a histogram by finding the pair of values that gives a best fit to the histogram in the least squares sense [C20].

c) Object identification using constraint filtering

The conventional approach to recognizing targets in FLIR imagery is to extract potential target regions using segmentation techniques, and then carefully analyze the properties of each region independently in order to determine whether or not it could be a target. We have investigated a complementary approach based on comparisons among regions rather than analysis of individual regions. After the image is segmented, we give each region a set of possible labels - e.g., "sky," "ground," "smoke," "tree," "tank." We then attempt to eliminate labels from the regions based on their relationships with other regions (relative property values, relative positions, etc.). This method performed successfully in a small set of tests; it eliminated the "tank" label from all the non-tank regions but kept it for all the tank regions [C25]. This approach should be of interest as a supplement to existing target recognition algorithms.

1.3. Feature extraction and background analysis

a) Edge and corner extraction

Feature detection (e.g., edge detection) is an important adjunct to object recognition, and also plays an important role in image matching (e.g., for object tracking and time-varying imagery analysis). Three feature detection studies were conducted on this project. The optimal approach to edge detection developed by Hueckel, which finds the best-fitting step function to a given image neighborhood, was applied to derive optimal edge operators for a class of small neighborhoods [C28]. A basic new method of evaluating edge detector output, based on consistency of the edge output data, was developed and successfully tested [C8]. A simplified method of corner detection was developed based on detecting discontinuities in one-dimensional projections of the image; this method eliminates the need to apply computationally expensive higher-order derivative operators at every point of the image [C13].

b) Blob and ribbon extraction

Work was also done on the detection of higher-level features such as "blobs" and "ribbons" in an image. (A blob is surrounded by consistently facing edges, while a ribbon is characterized by "antiparallel," oppositely facing edges.) Edge linking schemes were developed for detecting such features based on compatibility of the edges with respect to both geometry and gray level [C2]. Quantitative measures for edge compatibility were also developed for assessing both closedness [C3] and antiparallelness [C4].

c) Texture analysis

In connection with image background characterization, two texture analysis studies were conducted. An approach to texture analysis based on average strength of match with various local patterns was implemented; it was found to perform better than several standard methods [C18]. The idea of applying texture measures to arrays of terrain elevation data was also briefly explored; if such data were available at sufficient resolution, it would provide a useful supplement to intensity-based texture analysis [C15].

1.4. Multi-resolution image analysis

a) Background and related work

A potentially powerful new approach to image analysis, now under development at our laboratory, is based on the use of a "pyramid" of successively reduced-resolution versions of the given image. Initial work on image segmentation using pyramids was done under NSF sponsorship. During the summer of 1982, a workshop on "Multiresolution Image Processing and Analysis" was held, also under NSF sponsorship, at which about 25 research groups presented recent results that make use of multiresolution image representations in various ways. The pyramid image representation also has the advantage of compatibility with the quadtree region representation, which was extensively studied during an earlier phase of this project, and which is being further studied in connection with cartographic data base applications under the sponsorship of the U.S. Army Engineer Topographic Laboratory.

b) Segmentation and representation

One way of using the pyramid representation segmentation is to define links between pixels and their "parents" at consecutive levels of the pyramid, based on mutual similarity; this gives rise to subtrees of the pyramid, and thus defines a partition of the image, where each part consists of the pixels that are the leaves of a given subtree. A number of variations on this basic approach were investigated [C6], and it was also generalized to multispectral imagery [C11]. In connection with quadtree region representation, earlier work on the generation of an image row by row from its quadtree was extended to include several new algorithms [C7].

c) Feature extraction and encoding

Pyramids can also be used to extract and represent features such as edges and blobs in an image. If we use mutual similarity as a basic for linking "edgels," rather than pixels, in a pyramid representation, we obtain "trees" of edges which allow us to detect the major edges in an image, at the higher levels of the pyramid, and then locate these edges precisely at the full-resolution level [C14]. Pyramids can also be used to encode edges (or curves) detected in an image, yielding successively coarser approximations as long as the edges crossing a given block of the image can be compactly approximated [C5]. These approximations can then be used as an aid in linking together edge segments that lie on long lines or smooth curves [C30]. Two approaches to blob extraction using pyramids were also investigated. One of these uses pixel linking to construct subtrees of the pyramid such that leaves of each subtree are the pixels that belong to a compact, homogeneous piece of the image [C24]. Another approach is based on the fact that any blob shrinks to a (local) "spot" at some level of the pyramid; it detects blobs by constructing an edge pyramid and detecting pixels that are locally surrounded by edges [C21]. This method outperformed all the others that were tested in the comparative study of FLIR image segmentation techniques (see above, Section 1.2a).

1.5. Time-varying imagery analysis

a) Image matching

One approach to detecting and analyzing motion in an image sequence is to identify sets of corresponding points in successive frames of the sequence. This is usually done by searching for matches to pieces of one frame in the other frame. In order to obtain sharp matches, it is desirable to use pieces that contain distinctive, high-contrast features such as corners (they are preferable to edges because the match to an edge is insensitive to displacement in the direction along the edge). Some successful experiments in image matching using corner features are described in [C12]. A supplemental experiment, reported in [C17], showed that local intensity-based matching in the neighborhood of a feature point can be used to unambiguously locate match peaks in those cases where the results of the feature matching are ambiguous.

b) Motion estimation and smoothing

Another approach to motion detection, appropriate in cases where the rate of motion does not exceed one pixel per frame, involves using the space and time derivatives of the image intensity at each pixel to estimate a motion vector at that pixel. This method yields reliable estimates of motion components only in directions where there are rapid changes in gray level. Thus in a smooth region it yields no useful information; at an edge it yields only the component of motion in the direction across the edge; but at corner pixels it yields two components, thus allowing the entire motion vector to be estimated [C16]. Given a region in the image representing a rigid object moving parallel to the image plane, we can estimate motion vectors at the corners of the object and "propagate" these estimates around the edges of the object to determine its motion (translation and rotation). This approach to motion estimation was developed in a series of reports [C22,C23,C29].

The motion vector fields obtained from small image neighborhoods are noisy. If they are smoothed by simple local averaging, incorrect results are obtained at the boundaries of moving objects. A better approach is to use nonlinear smoothing techniques based on selective local averaging; this does not blur sharp edges [C31]. A related problem is that of smoothing the images in a sequence by averaging successive frames; here one cannot simply average corresponding pixels, but must introduce displacements in order to allow for the motion. In this connection, one need not know the entire motion vector, but only its component in the gradient direction, since errors in the tangential direction will not cause edges to become blurred [C32].

c) Optical flow analysis

The changes in an image sequence due to the motion of the observer relative to the scene, rather than to object motion, are known as "optical flow." Given an array of motion vectors representing optical flow, methods have been developed of inferring the parameters of the observer's motion (translational and rotational) and of deriving the relative distances between the observer and the points in the scene. Algorithms for deriving relative scene distance and local surface orientation from optical flow are presented in [C9], while a method of deriving the observer's instantaneous direction of motion from optical flow, and of decomposing his motion into translational and rotational components, is developed in [C10].

1.6. Status reports

As mentioned in the Introduction, three project status reports were issued [B1-3] summarizing the work done during this phase of the project. The first and third of these reports were also published in the Proceedings of the two DARPA Image Understanding Workshops that were held during this period (April 1981 and September 1982).

At a meeting of the principal investigators on the DARPA Image Understanding Program, held in January 1982, it was decided to prepare a Final Report on the overall program. The University of Maryland was asked to draft the portion of this report dealing with two-dimensional image analysis techniques ("low-level vision"). An edited version of this draft was also issued as a technical report [C27].

2. AUTONOMOUS VEHICLE NAVIGATION

This project is concerned with developing navigation techniques for an autonomous outdoor ground vehicle. The vehicle will have access to a stored database containing information about the terrain on which it is to operate, and will have sensory input from a passive optical or IR sensor. The key problem in navigating the vehicle is to relate the sensory input to the stored data in order to determine the location of the vehicle and the locations of landmarks or goals, and to plan paths (from the current location to a goal) that satisfy given constraints. Additional tasks, on which preliminary work will also be done, relate to short-range sensing (e.g., for obstacle avoidance) and to real-time analysis of time-varying imagery.

Since this project was initiated quite recently, this report provides only a general outline of the planned tasks. A vehicle and a test site have been tentatively selected. Westinghouse will gather data regarding the site (e.g., high-resolution terrain model and sample imagery) and will also design and assemble the vehicle system. Maryland will develop algorithms for processing the imagery, relating it to the stored data, and planning paths for the vehicle. When these algorithms have achieved adequate performance, Westinghouse will adapt them to run on the vehicle's on-board computer, after which they will be tested under real-world conditions. Concurrently, Maryland will continue to study problems related to short-range sensing and real-time processing. Maryland's work during the initial months of the project has dealt primarily with time-varying imagery analysis; a paper reporting on one aspect of this work appears elsewhere in these Proceedings.

REFERENCES

A. Final Reports on previous phases of the project

1. Final report: Algorithms and hardware technology for image recognition, March 1978.
2. Final report: Image understanding using overlays, May 1980.

B. Interim Reports on the current phase of the project

1. Semiannual report (1 April 1980 - 31 January 1981), February 1981.
2. Semi-annual report (1 February - 31 July 1981), August 1981.
3. Project status report (1 August 1981 - 31 July 1982), July 1982.

C. Technical Reports on the current phase of the project

1. Alan Broder and Azriel Rosenfeld, "Gradient Magnitude as an Aid in Color Pixel Classification". TR-906, June 1980.
2. Mohamad Tavakoli and Azriel Rosenfeld, "Toward the Recognition of Buildings and Roads on Aerial Photographs". TR-913, July 1980.
3. Mohamad Tavakoli and Azriel Rosenfeld, "A Method for Linking Pairs of Compatible Linear Features". TR-930, August 1980.
4. Mohamad Tavakoli and Azriel Rosenfeld, "A Method for Finding Pairs of Anti-Parallel Linear Features". TR-943, September 1980.
5. Michael Shneier, "Two Hierarchical Linear Feature Representations: Edge Pyramids and Edge Quadtrees". TR-961, October 1980.
6. Teresa Silberberg, Shmuel Peleg, and Azriel Rosenfeld, "Multi-Resolution Pixel Linking for Image Smoothing and Segmentation". TR-977, November 1980.
7. Hanan Samet, "Algorithms for the Conversion of Quadtrees to Rasters". TR-979, November 1980.
8. Les Kitchen and Azriel Rosenfeld, "Edge Evaluation Using Local Edge Coherence". TR-981, December 1980.
9. K. Prazdny, "Relative Depth and Local Surface Orientation from Image Motions". TR-996, January 1981.
10. K. Prazdny, "Determining the Instantaneous Direction of Motion from Optical Flow Generated by a Curvilinearly Moving Observer". TR-1009, February 1981.
11. Tsai-Hong Hong and Azriel Rosenfeld, "Multiband Pyramid Linking". TR-1025, March 1981.
12. Cheng-ye Wang, Hanfang Sun, Shiro Yada, and Azriel Rosenfeld, "Some Experiments in Relaxation Image Matching Using Corner Features". TR-1071, July 1981.
13. Zhong-Quan Wu and Azriel Rosenfeld, "Filtered Projections as an Aid in Corner Detection". TR-1078, July 1981.
14. T. H. Hong, M. Shneier, and A. Rosenfeld, "Border Extraction Using Linked Edge Pyramids". TR-1080, July 1981.

15. Cheng-Ye Wang and Azriel Rosenfeld, "Elevation Texture". TR-1086, August 1981.

16. K. Prazdny, "Computing Motions of (Locally) Planar Surfaces from Spatio-Temporal Changes in Image Brightness: A Note". TR-1090, August 1981.

17. Hanfang Sun, "Image Registration by Combining Feature Matching and Gray Level Correlation". TR-1091, August 1981.

18. Matti Pietikäinen, Azriel Rosenfeld, and Larry S. Davis, "Texture Classification Using Averages of Local Pattern Matches". TR-1098, September 1981.

19. Ralph L. Hartley, Leslie J. Kitchen, Cheng-Ye Wang, and Azriel Rosenfeld, "A Comparative Study of Segmentation Algorithms for FLIR Images". TR-1104, September 1981.

20. Stanley Dunn, Ludvik Janos, and Azriel Rosenfeld, "Bimean Clustering". TR-1106, September 1981.

21. T. H. Hong and M. Shneier, "Extracting Compact Objects Using Linked Pyramids". TR-1123, November 1981.

22. Larry S. Davis, Hanfang Sun, and Zhongquan Wu, "Motion Detection at Corners". TR-1130, December 1981.

23. Zhongquan Wu, Hanfang Sun, and Larry S. Davis, "Determining Velocities by Propagation". TR-1132, December 1981.

24. Tsai Hong Hong and Azriel Rosenfeld, "Unforced Image Partitioning by Weighted Pyramid Linking". TR-1137, January 1982.

25. Les Kitchen, "Scene Analysis Using Region-Based Constraint Filtering". TR-1150, February 1982.

26. Cheng-Ye Wang and Leslie Kitchen, "Improvements in Multispectral Image Smoothing". TR-1152, March 1982.

27. Azriel Rosenfeld, "Computer Vision". TR-1157, April 1982.

28. Ravi B. Boppana and Azriel Rosenfeld, "Some Properties of Hueckel-Type Edge Operators". TR-1178, June 1982.

29. Larry S. Davis, Zhongquan Wu and Hanfang Sun, "Contour-based Motion Estimation". TR-1179, June 1982.

30. T. H. Hong, M. Shneier, R. Hartley and A. Rosenfeld, "Using Pyramids to Detect Good Continuation". TR-1185, July 1982.

31. Kwangyoen Wohn, Larry S. Davis and Philip Thrift, "Motion Estimation Based on Multiple Local Constraints and Nonlinear Smoothing". TR-1188, August 1982.

32. Larry S. Davis, Hu-chen Xie and Azriel Rosenfeld, "Image Sequence Enhancement Based on the Normal Component of Image Motion". TR-1189, July 1982.

IMAGE UNDERSTANDING RESEARCH AT THE UNIVERSITY OF MASSACHUSETTS

Edward M. Riseman and Allen R. Hanson

Computer and Information Science Department
University of Massachusetts
Amherst, Massachusetts 01003

The major focus of the our DARPA funded research program revolves around issues of dynamic image processing. The authors We have been examining techniques for recovery of environmental information, such as depth maps of the visible surfaces, from a sequence of images produced by a sensor in motion. Algorithms that appear robust have been developed for constrained sensor motion such as pure translation, pure rotation, and motion constrained to a plane. Interesting algorithms with promising preliminary experimental results have also been developed for the case of general sensor motion in images where there are several significant depth discontinuities, and for scenes with multiple independently moving objects. A general hierarchical parallel algorithm for efficient feature matching has also been developed for applications of motion, stereo, and image registration.

In addition, they we have been designing a highly parallel architecture that integrates aspects of both parallel array processing and associative memories for real-time implementation of motion algorithms. Finally, there has been a continuation of the VISIONS static image interpretation project, with interesting results in top-down processing of a set of complex outdoor house scenes. Each of the above research topics is documented in papers appearing in these proceedings [1-6].

I. QUANTITATIVE MOTION PROCESSING FOR RECOVERY OF ENVIRONMENTAL DEPTH

I.1. INTRODUCTION

The major goal in motion processing is the recovery of the motion parameters of the sensor and each independently moving object. The computation of environmental depth of visible surfaces follows in a rather straightforward manner. This has generally involved two stages of processing: computation of a feature displacement field, followed by inference of motion parameters and environmental depth. We will present several algorithms for performing this computation in independent stages, and in several restricted cases of sensor motion some new alternatives for combining the two stages in a robust manner.

The set of image displacements from two or more images is an approximation to optic flow. During this stage of the processing one faces the well-known correspondence problem, which involves the matching of corresponding image points of an environmental feature in the pair of images. The second stage involves inference of environmental information from the optic flow or the displacement field. This becomes a problem of separating the translational and rotational components of the flow field.

Rotation of the sensor induces image displacements that are a function only of the rotational parameters and image position; in particular the feature displacement between images is not a function of the depth of its environmental surface point.

The translational motion of the sensor carries all of the environmental cues. For purely translational motion, the image displacement paths are determined by radial flow lines emanating from a single point in the image plane, that is the intersection of the translational axis with the image plane (also referred to as the focus of expansion - FOE). The size of displacements along these paths are a function of environmental depth and distance from the FOE. Thus, the problem of general motion becomes one of decomposing the rotational and translational effects of motion, and then using the image displacements from the instantaneous component of translational motion to compute depth.

I.2. RESTRICTED CASES OF SENSOR MOTION

Our primary technique for depth inference has been derived in Lawton's forthcoming doctoral dissertation [7]. He has shown that in the cases of restricted sensor motion - pure translation, pure rotation, and motion constrained to a plane - one can bypass, or at least simplify, the correspondence problem by combining the computation of the motion parameters with the determination of image displacements.

Let us illustrate with the case of pure translational motion [3]. There are two unknown sensor parameters which can be specified by the intersection of the translation axis with the image plane (the FOE). For a given FOE, the flow lines emanate radially from this point, and therefore the matching of an image point in one frame to its new position in the second frame has been reduced to a one-dimensional search along the straight line between the FOE and the image point. While there may still be spurious high correlations possible, the number of incorrect good matches will be greatly reduced over the usual two-dimensional correlation process. In cases of the incorrect FOE there is a strong probability that many points will have poor correlations at all points along the hypothesized displacement path. The shape of the resulting error function can be improved by selection of "interesting" image points of high contrast (boundaries) and high curvature (corners).

The determination of the translational motion parameters has now become a search process using a global error measure which is the sum of the errors of the best match on each point's flow path. The search process consists of two phases: a global sampling of the error measure, and then a local search at a finer sampling to determine the minimum. The error function appears to be very well behaved in a series of experiments on real scenes, and the algorithm seems rather robust.

In the case of pure rotation, the basic technique can be applied with minor differences. The search space for the correct rotational parameters is three-dimensional: two parameters for the axis of rotation and one for the magnitude of rotation. The algorithm can proceed in the same manner by choosing a set of distinguished points, and then compute a global error on a coarsely sampled parameter space. This problem is actually slightly more constrained than the first, because here the third dimension of the amount of rotation will directly constrain the image motion of all points simultaneously, while in the translational case each point had to be matched independently (because of differences in environmental depth).

In the case of motion restricted to a known plane, there are only two degrees of freedom. Translational motion will be constrained to the one dimension of the line represented by the intersection of the known plane and the image plane. The axis of rotation must be perpendicular to the plane, and therefore we must only determine the degree of rotation.

A set of experiments have proven these algorithms to be very robust in real scenes, including the outdoor road sequence from William's thesis [10] and industrial image domains supplied by the General Electric Corporation.

I.3. RECOVERY OF DEPTH FROM GENERAL SENSOR MOTION

As we have pointed out earlier, the flow fields produced by a sensor undergoing general motion are difficult to interpret until they have been decomposed into their rotational and translational components. Once this has taken place, environmental depth can be recovered from translational displacements. Analytical techniques for performing this computation are extremely complex and can be quite sensitive to the errors that are typical in the computation of displacement fields. It is not feasible to exploit the approach of the previous cases where potential motion parameters were tested by computing a global error measure of lack of consistency across a set of image features. In the previous cases the dimensionality of the search space was no greater than three, but here it is a five-dimensional search space, and the computational demands may be excessive. In addition the error function cannot be expected to be well-behaved so that simple optimization techniques probably would not work.

Recently Lawton and Rieger [2] have described a surprisingly simple technique that promises to be rather robust in noisy, low resolution and/or sparse displacement fields. It depends upon the scene containing a sufficient number of depth discontinuities of sufficient depth difference. Thus, a scene with several objects at distinct depths, or a single object of reasonable size against a textured background, will permit this technique to be effective.

Consider distinct surface features at different depths on an occlusion boundary. Sensor rotation causes an equal rotational displacement because these points appear at the same image location. Thus, the only difference in their image displacement is caused by a difference in translational displacement. This leads to an algorithm which will exploit nearby image points which are at different depths. Note, however, that occlusion need not be determined because differences can be taken of all nearby flow vectors. They will be oriented on radial flow lines, emanating from the instantaneous axis of translation which can be determined by an optimization procedure. There are several approaches to determining the axis of translation, such as the use of a Hough transform to select the point that most nearly lies at the intersection of these vectors. Due to practical noise considerations, a global error measure is used to evaluate each possible value for the direction of the translational axis in a coarse to fine search. The error measure used is the sum of the magnitudes of the error angles of the difference vector field and the set of radial field lines. Once the instantaneous axis of translation is determined, then the rotational component is overconstrained, can be determined and then subtracted out. Environmental depth of image points can then be computed from the translational displacement.

The algorithm is not quite so straightforward because there may not be many reliable image displacement vectors that are at different depths and near each other. To the degree that they are not at sufficiently different depths, their difference vector will be short and prone to error. To the degree that they are not near each other, their rotational components will differ and introduce error. Thus, practical considerations in the application of the algorithm remain. However, several experiments have shown very promising results.

It should be noted that occlusion boundaries of independently moving objects will not satisfy the conditions for applying this algorithm, and thus the next algorithm complements this work.

I.4. SCENES WITH MULTIPLE INDEPENDENTLY MOVING OBJECTS

The algorithms that we have just described do not confront the additional complexity introduced when there are multiple independently moving objects. The global types of constraints that were described earlier no longer apply across the entire image. The case of a sensor moving through a static environment can be equivalently viewed as an image of a single rigid object with associated motion parameters. However, if there are independently moving objects, they will have different motion constraints and introduce possibly serious errors in the global search of the parameter space for a single set of motion parameters. Thus, the goal is to decompose the image, and thereby separate the information in each flow field, so that motion of each object can be recovered.

The approach outlined here is presented by Adiv [4]. It involves a generalized Hough transform, proposing solutions to some of the problems found in this technique. Hough techniques are relatively insensitive to noise and can deal with partially incorrect or occluded data. Here, such a transform will be used to group a set of displacement vectors which satisfy the same motion parameters. However, there are a set of problems that must be considered: non-adjacent elements can vote for the same image transformation, there are difficulties in the detection of the motion parameters of small objects, and fine resolution of the motion parameter space can require large amounts of memory and computation time.

The suggested solution to these problems involves a modified multipass approach. In each pass windows are located around potential objects by the degree to which the displacement field is locally inconsistent with previously found motion transformations. The Hough transform is applied separately to the displacement vectors in each window. Thus, the sensitivity of the Hough transform to local events is increased and the motion parameters of small objects can be detected even in a noisy displacement field. A multiresolution scheme in both the image plane and the parameter space reduce the computational cost, while still maintaining accuracy.

The algorithm has been shown to be efficient and robust in extracting motion parameters from artificial images with objects undergoing 2D motion. It involves a 4-dimensional parameter space of horizontal translation, vertical translation, rotation (in the image plane) and expansion/contraction.

The current research invovles the extension of this approach to 3D motion and to real scenes. This extension is non-trivial because displacement vectors in the 2D motion case involve four parameters with two constraints; thus, each displacement vector "votes" for a two-dimensional hyperplane of the parameter space. In the case of 3D motion when surface depth is unknown, there will be 5 motion parameters, and each displacement vector provides only one constraint; i.e., each will vote for a four-dimensional subspace of parameter cells. Thus, the signal to noise ratio in the parameter space will be much lower, and with the presence of noise in real images, the determination of peaks in generalized Hough space will be challenging.

II. FEATURE MATCHING BY HIERARCHICAL CORRELATION

Feature matching algorithms are important in problems involving motion detection, image registration, and stereo vision. Hierarchical correlation provides a computationally efficient feature matching strategy. They can be implemented in hierarchical parallel hardware architectures, and they can also be implemented on a sequential machine to run very efficiently using a coarse to fine matching strategy.

Glazer, Reynolds, and Anandan [4] have developed a hierarchical matching algorithm that consists of matching band-passed versions of the images at different levels of resolution. The filters approximate convolution of a Laplacian and a Gaussian (del-squared-G) of different sizes. Alternative computational techniques for implementing the band-pass filter are being examined. One technique involves computing the del-squared-G at the finest level followed by a 4x4 Gaussian centered on 2x2 windows to reduce the resolution by a factor of two on each axis. These algorithms are computed in the processing cone [8] of the VISIONS Image Operating System [9].

The matching is performed first on the low frequency structures occurring at the coarsest levels of the images, thus providing a coarse to fine strategy for matching higher frequency information at the levels below. This reduces the problem of false matches when, for example, there is high frequency texture with somewhat repetitive patterns. Thus all useful information of the image is utilized at different levels: low frequency information at coarser levels and higher frequency information at finer levels.

The correlation strategy utilizes the observation that at some sufficiently coarse level, the maximum displacement of an image event between a pair of images is at most one pixel. This restricts the search at that level to a 3x3 area and provides an estimate of displacement within ± 1/2 pixel accuracy. The projection of this estimate to the next finer level provides an estimated displacement of ± 1 pixel and allows search to again be restricted to a 3x3 area, with the process repeating downward. There are two significant computational advantages of this process. The number of correlation matches considered is 9*logD instead of (2D+1)**2, where D is the maximum displacement possible at the finest level of resolution. In addition, an 8x8 correlation window size was used at all levels, and this would require a window of size (8D)**2 to capture the same amount of information in a single level of search across correlation positions.

The algorithm has shown in practical experiments to be effective in determining even small amounts of rotation, seems to be insensitive to noise, and of course is very efficient. Experiments have shown that it may not be necessary to apply the algorithm to restricted sets of interesting points that have a high degree of distinctiveness (such as corners). Some experiments have shown consistently correct results using all points, and thus might work on an arbitrary sampling of points.

III. A CONTENT ADDRESSABLE ARRAY PARALLEL PROCESSOR (CAAPP)

Our research environment has maintained a continuous interest in parallel architectures and parallel algorithms. Real-time motion processing will require between one and two orders of magnitude more computational power than static vision. Thus, VLSI technology and massively parallel machines are obvious research directions.

Weems, Levitan, and Foster [11] have developed a design for a Content Addressable Array Parallel Processor (CAAPP) and have been applying it to the motion algorithms with Lawton [5]. The CAAPP is both a 512x512 Single Instruction Multiple Data (SIMD) array processor and an associative memory. The design is based on a 64x64 array of custom VLSI chips and is intended to act as a slave processor for a general purpose computer system. Each chip then contains 64 cells, an instruction decoder, and some miscellaneous logic. There are eight basic instruction types recognized by the chip, each performed in parallel by the constituent cells. Most instructions take one minor cycle time (100 nanoseconds) to execute. Inter-cell communication is bit serial and is accomplished by a four-way (N, S, E, W) cell interconnect network, allowing for three types of edge treatments: dead-edging, circular wrap, and zig-zag wrap. The entire memory may be bulk-loaded in one video frame time (1/30 second).

A very interesting application developed for the CAAPP (that makes use of the associativity and array processing capabilities) is an effective means of quickly and accurately decomposing a flow field into its rotational and translational components to recover the parameters of sensor motion. The algorithm is an exhaustive search procedure via a top-down parallel correlation of a set of rotational and translational flow field templates to find a component pair which most closely accounts for the motion depicted in a given flow field. Currently, 1000 rotational templates and 200 translational templates are used. Each cell contains the horizontal and vertical components of a flow vector, each specified with 10 bits of precision.

Experiments have been performed with a CAAPP simulator on a VAX 11/780 using a wide variety of motions and simulated environments. In all cases examined, the translational template closest to the actual translational motion was selected. The rotational template was always close to the actual rotational motion, but was sometimes not the closest template. The procedure proved to be resistant to limited Gaussian noise as well as to limited random spike noise in the original flow field. The CAAPP timing calculations revealed that the algorithm could perform the rotational-translational decomposition in slightly more than 1/4 second. Given fabrication techniques available in the immediate future, execution times can be expected to be significantly improved.

Using the CAAPP strictly as a parallel array processor it is of course possible to perform standard image processing operations such as convolution. For example, a simple 3x3 Gaussian mask convolution can be done in 98 microseconds on the CAAPP. It should be noted that the time required to perform a convolution on the CAAPP is constant for a given image size and only varies depending on the size and complexity of the mask. For example, a 10x10 mask of 8 bit multipliers applied to an image of 16 bit pixels (with the same number of pixels as the previous example) would require on average approximately 30 milliseconds (about one frame time). The method used is not restricted to square masks and is actually easily adapted to such shapes as annuli and disjoint areas.

IV. RULE BASED STRATEGIES FOR IMAGE INTERPRETATION

As part of the VISIONS long-term project for the interpretation of static images, we have developed an experimental testbed for examining issues in knowledge directed processing. Weymouth, Griffith, Hanson and Riseman [6] have been developing a rule based image interpretation system which has been effective in interpreting a set of complex outdoor scenes. The system utilizes world knowledge in the form of simple object hypothesis rules, and more complex interpretation strategies attached to object and scene schema, to reduce the ambiguities in image measurements.

Descriptions of scenes, at various levels of detail, are stored in a set of schema hierarchies [12]. A schema graph is a data structure defining an expected collection of objects, such as a house scene, the expected visual attributes associated with the objects in the schema (each of which can have an associated schema), and the expected relations among them. This stored knowledge can be used to infer the presence and location of other objects, or verify uncertain hypotheses via spatial consistency of object labels. However, in order to use this knowledge there must be a basis for partial interpretations.

In the initial stages, there are few if any image hypotheses, and development of a partial interpretation must rely primarily on general knowledge of expected object characteristics that are independent of other hypotheses. We propose an approach to object hypothesis formation which is both simple and effective. It relies on convergent evidence from a variety of measurements and expectations. The rules involve sets of partially redundant features each of which defines an area of feature space which represents a "vote" for an object. The features include color, texture, shape, size, image location, and relative location to other objects. For example, in an outdoor scene taken with a camera in standard position, one would expect grass to be of medium brightness, to have a significant green component, to embody a modest degree of texture, to be located somewhere in the lower portion of the image, etc. These expectations are translated into a rule which combines the results of many measurements into a confidence level that the region (or group of regions) represents grass.

Convergent evidence from multiple interpretation strategies is organized by top-down control mechanisms in the context of a partial interpretation. The extreme variations that occur across images can be compensated for somewhat by utilizing an adaptive strategy. This approach is based on the observation that the variation in the appearance of objects (region feature measures across images) is much greater than object variations within an image. One such strategy extends a kernel interpretation derived through the selection of object exemplars, which are regions that represent the most reliable image specific hypotheses of a general object class. The use of exemplar strategies and other top-down strategies results in the extension of partial interpretations from islands of reliability. Finally a verification phase can be applied where relations between object hypotheses are examined for consistency. Thus, the interpretation is extended through matching and processing of region characteristics as well as semantic inference.

Experiments are being conducted on a set of fifteen "house scene" images. Thus far, we have been able to extract sky, grass, and foliage (often separating trees and bushes) from nine house images with reasonable effectiveness, and have been successful in identifying houses and their parts, including shutters (or windows), house wall and roof in three of these images. The interpretation strategies use many redundant features, each of which can very often be expected to be present. The premise is that many redundant features allow any single feature to be unreliable. The features utilized include those mentioned earlier (color and texture attributes, shape, size, location in the image), as well as relative location to identified objects, and similarity in color and texture to identified objects. Object hypothesis rules were employed as described in previous sections, and additional object verification rules requiring consistent relationships with other object labels are being developed.

REFERENCES

[1] Lawton, D., "Processing Restricted Motion," (these proceedings).

[2] Lawton, D. and Rieger, J., "The Use of Difference Fields in Processing Sensor Motion," (these proceedings).

[3] Adiv, G., "Recovering 2-D Motion Parameters in Scenes Containing Multiple Moving Objects," (these proceedings).

[4] Glazer, F., Reynolds, G. and Anandan, P., "Scene Matching by Hierarchical Correlation," (these proceedings).

[5] Weems, C., Levitan, S., Lawton, D. and Foster, C., "A Content Addressable Array Parallel Processor and Some Applications," (these proceedings).

[6] Weymouth, T., Griffith, J., Hanson, A. and Riseman, E., "Rule Based Strategies for Image Interpretation," (these proceedings).

[7] Lawton, D., "Processing Dynamic Image Sequences from a Moving Sensor," Ph.D. Thesis, Computer and Information Science Department, University of Massachusetts at Amherst, in preparation, 1983.

[8] Hanson, A. and Riseman, E., "Processing Cones: A Computational Structure for Image Analysis," in Structured Computer Vision (S. Tanimoto, ed.), Academic Press, New York, 1980. Also COINS Technical Report 81-38, University of Massachusetts at Amherst, December 1981.

[9] Kohler, R. and Hanson, A., "The VISIONS Image Operating System," Proc. of 6th International Conference on Pattern Recognition, Munich, Germany, October 1982.

[10] Williams, T., "Computer Interpretation of a Dynamic Image from a Moving Vehicle," Ph.D. Thesis and COINS Technical Report 81-22, University of Massachusetts at Amherst, May 1981.

[11] Weems, C., Levitan, S. and Foster, C., "Titanic: A VLSI-Based Content Addressable Parallel Array Processor," Proc. of IEEE International Conference on Circuits and Computers, September 1982, pp. 236-239.

[12] Hanson, A. and Riseman, E., "VISIONS: A Computer System for Interpreting Scenes," in Computer Vision Systems (A. Hanson and E. Riseman, eds.), Academic Press, 1978, pp. 303-333.

Dup

Recent Results of the Rochester Image Understanding Project

J.A. Feldman, D.H. Ballard and C.M. Brown

Computer Science Department
University of Rochester
Rochester, New York 14627

1. Robust Vision Operators

1.1. Parameter Networks and the Hough Transform

One of the most difficult problems in vision is segmentation. Recent work has shown how to calculate intrinsic images (e.g., optical flow, surface orientation, occluding contour, and disparity). These images are distinctly easier to segment than the original intensity images. Such techniques can be greatly improved by incorporating Hough methods. The Hough transform idea has been developed into a general control technique. Intrinsic image points are mapped (many to one) into 'parameter networks' [Ballard, 1983]. This theory explains segmentation in terms of highly parallel cooperative computation among intrinsic images and a set of parameter spaces at different levels of abstraction.

The most recent application of these ideas are to improved shape-from-shading calculations which work on several spaces [Brown et al., 1983] and motion extraction [Ballard & Kimball, 1983]. This domain specific effort is closely linked to our new work on a more general theory of Hough-like computations and general implementation techniques for them.

The theory is also useful in analysis of cache-based Hough Transform implementations. It is an appealing idea to use a small content-addressable store to accumulate Hough transform results, rather than a potentially huge multi-dimensional array. The initial technical issues were discussed in [Brown & Sher, 1982]. We are currently pursuing VLSI implementations.

1.2 Hough Transform Implementation

Earlier work on the Hough transform [Brown, 1983; Brown & Sher, 1982] has led in three directions.

1) Research toward a theory of cache accumulator arrays [Loui, 1983; Brown & Feldman, 1983]

2) Experiments with complementary HT and cache management strategies [Brown et al., 1983]

3) Hardware (VLSI) designs for HT vote caches [Sher & Tevanian, 1983].

Work in each of these directions is in progress; some of the cited references are draft documents. The behavior of caching schemes for accumulation of votes in the Hough transform is equivalent to the statistical problem of estimating the mode of a distribution using only a finite memory for vote tallies, and is a generalization of the familiar 'secretary' ('maximum of a sequence,' 'beauty contest') problem. Loui's document explores this avenue for analysis. The experiments with HT implementation are to see how well the peak-sharpening provided by complementary HT performs with real images on complex shapes. Work on cache architectures (hierarchical schemes, cascaded caches) is ongoing.

The VLSI design project produced a circuit for vote cacheing that can be cascaded to provide a cache of any length. Work on improving the efficiency and power of the design will continue this summer.

1.3 High Level Planning

In general, problem solvers cannot hope to create plans that are able to specify fully all the details of operation beforehand and must depend on run-time modification of the plan to insure correct functioning. The run-time planning idea becomes particularly important when different plan segments are being explored concurrently. These communicating segments may require sophisticated actions e.g. (do $PLAN_x$ until $PLAN_y$). These issues are being studied by [Russell] in the context of a cooperative planning and execution system for manipulation tasks.

2. Computing with Connections

We are continuing our interest in problem-scale parallelism, both as a model of animal brains and as a paradigm for VLSI. Work at Rochester has concentrated on connectionist models and their application to vision. The framework is built around computational modules, the simplest of which are termed p-units. We have developed their properties and shown how they can be applied to a variety of problems [Feldman & Ballard, 1982]. More recently, we have established powerful techniques for adaptation and change in these networks [Feldman, 1982].

A major milestone was achieved with Sabbah's thesis on massively parallel recognition of Origami-world objects [Sabbah, 1982]. Sabbah's work extended the connectionist methodology to a problem domain with several hierarchical structural levels. The resulting program is, to our knowledge, the most noise-resistant system for dealing with this level of complexity. One outcome of Sabbah's effort has been a project to build a general purpose simulator for massively parallel systems [Small et al., 1982].

The general connectionist simulator has been well tested and is being used in a number of applications. One project involves a quite detailed simulation of motor control networks of the occulo-motor system [Addanki, 1983]. Another application is to a spreading activation model of word sense disambiguation and related problems in natural language understanding [Cottrell & Small, 1983]. A major new effort involves modelling conceptual knowledge (such as that needed for high level vision) in connectionist terms. The simulator has also been a starting point for some of our efforts towards VLSI realization of connectionist machines (Section 5).

For a VLSI design couse, a circuit was designed to implement key aspects of the "connectionist" computational paradigm [Rainero & Kautz, 1983]. This cited document is a course project report, and the exercise was mainly useful in isolating particular technical problems that must be addressed in any such parallel, activation-passing computer.

3. Motion

Our interest in motion has centered around methods for extracting rigid body parameters from optic flow and intensity images. These parameters are extremely useful in navigation and target tracking. Currently these nine parameters (origin, translational velocity, rotational velocity) can be extracted from flow via a Hough technique [Ballard & Kimball, 1983]. We are also pursuing the use of these parameters to speed up the flow computations themselves [Stuth et al., 1983].

4. Shape

The description and recognition of complex shapes continues to be a major focus of the project. The analysis of the dot product space representation has been improved to handle certain pathological cases, and has been generalized to accommodate different criteria for the goodness of the representation.

This simple concept of shape has been applied to the problem of reconstructing three-dimensional surfaces from very sparse data. The key idea is to use appropriate shape descriptors to hypothesize a transformation which accounts for the difference in shape between successive contours. When the hypothesized transformation is minor, very simple-minded surface reconstruction techniques are sufficient. When there are major differences in shape or position between successive contours, our method hallucinates new contours, using the hypothesized shape transformation [Sloan & Hrechanyk, 1981].

Hierarchical descriptions of shapes were considered in [Ballard & Sabbah, 1981] in a preliminary fashion. Our previously reported shape model [Hrechanyk & Ballard, 1982] concentrated on problems of view-invariance and attention shifting within a single prototype. This model has been extended to handle the problems of extracting primitive shape descriptions from noisy images. Our work was motivated by dissatisfictions with smoothness criteria for intrinsic image conputations. Our current model uses correspondences called view frames as primitives. This model allows gestalt grouping to be modelled as well as parallel search for prototypes and parameter tracking [Hrechanyk & Ballard, (this Proceeding)].

The practicality of shape from shading computations and their interaction with the determination of other image parameters (such as illuminant position) was addressed by two papers in the Fall, 1982 DARPA Image Understanding Workshop. Since then we have implemented a multi-resolution shape-from-shading algorithm that exhibits high efficiency and accuracy in surface reconstruction of large (128 x 128) irregular shapes (Figure 1). We are now applying the algorithm to real images, and want to investigate scenes with non-Lambertian reflectance functions that are unknown apriori. We want to explain how humans in fact use shading to derive shape, given the complexity of reflectance functions and imaging situations in the world. Two competing theories are that somehow the reflectance functions are derived fairly accurately by an adaptive procedure, or instead that we only 'support' a small number of reflectance functions that are selected by other cues (such as gloss).

5. General Theory of Vision

Work in our laboratory, among others, has demonstrated strong links between powerful IU techniques and computations used by animal visual systems. We have established strong ties with a wide range of visual scientists at Rochester and a variety of collaborative efforts are underway. One early project is to survey the computational similarities in natural and computer vision [Ballard & Coleman, 1983].

We have begun to exploit Rochester neurobiology expertise in order to hone and improve our connectionist modelling efforts. One difficult avenue is to specify the interface between our computational models and the state-of-the-art neurobiological picture. Our efforts in this direction are summarized in [Ballard & Coleman, 1983] and the collaboration is continuing. Another effort is our attempt to develop a general framework for theories of vision that would provide a common structure for integrating studies from various disciplines [Feldman, 1982].

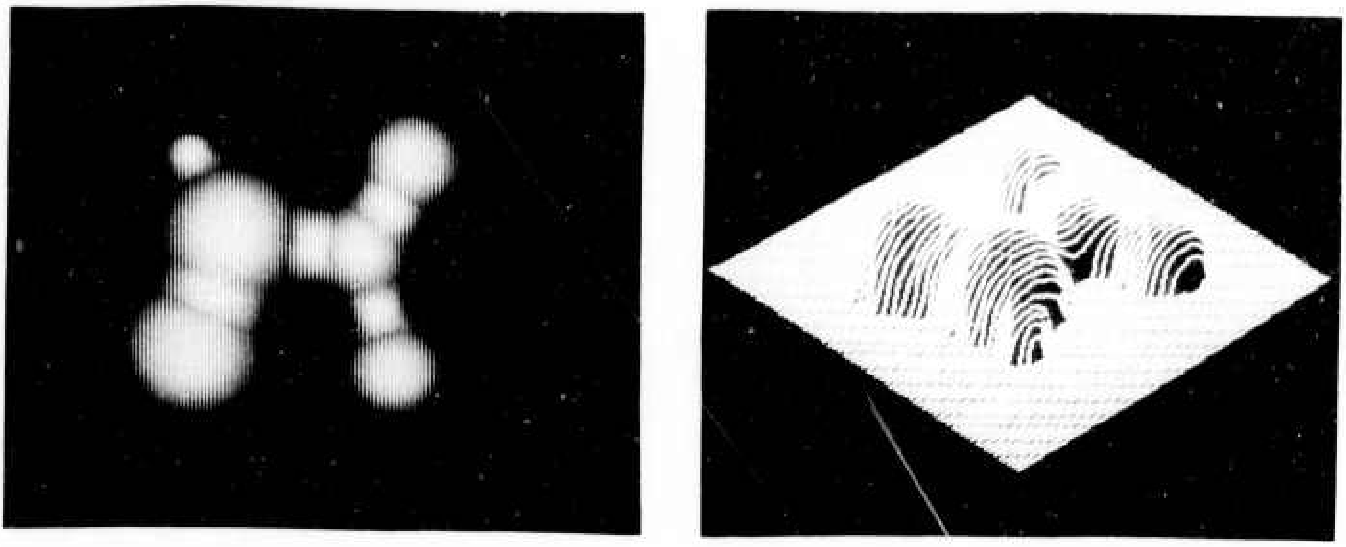


Figure 1: Surface Reconstruction of Large Irregular Shapes

References

Addanki, S., "On a Distributed Approach to Occulomotor Control," TR121, Computer Science Dept., University of Rochester, 1983.

Ballard, D.H., "Parameter Networks: Towards a Theory of Low-Level Vision," *Proceedings*, 7th IJCAI, Vancouver, British Columbia, August 1981.

Ballard, D.H. and P.D. Coleman, "Cortical Connections: Structure and Function," presented at U. Mass. Workshop on Vision, Brain and Cooperative Computation, Amherst, Mass., May 1983.

Ballard, D.H. and O.A. Kimball, "Rigid Body Motion from Depth and Optical Flow," *CVGIP* Special Issue on Computer Vision, 1983; also TR70, Computer Science Department, University of Rochester, November 1981.

Ballard, D.H. and D. Sabbah, "Detecting Object Orientation from Surface Normals," *Proceedings*, 7th IJCAI, Vancouver, British Columbia, August 1981.

Brown, C.M., "Bias and Noise in Hough Transform: Theory," to appear, *Pattern Analysis and Machine Intelligence*, 1983; also, TR105, Computer Science Department, University of Rochester, July 1982.

Brown, C.M., M. Curtiss, and D. Sher, "Bias & Noise in Hough Transform: Experiments," *Proceedings*, IJCAI-83, Karlsruhe, West Germany, August, 1983; also TR113, Computer Science Dept., University of Rochester, 1982.

Brown, C.M. and D. Sher, "Modeling the Sequential Behavior of Hough Transform Schemes," *Proceedings*, DARPA Image Understanding Workshop, November, 1982; also TR114, Computer Science Department, University of Rochester, August 1982.

Brown, C.M. and J.A. Feldman, "Statistical Questions Arising in the Use of Hough Techniques in Image Understanding," *Proceedings*, ONR Workshop on Statistical Image Processing and Graphics Workshop, Luray, VA., 24-27 May 1983.

Cottrell, G.W. and S.L. Small, "A Connectionist Scheme for Modelling Word Sense Disambiguation," *Cognition and Brain Theory*, 6 (1), 89-120, 1983.

Feldman, J.A., "A Connectionist Model of Visual Memory," in *Parallel Models of Associative Memory*, G.E. Hinton and J.A. Anderson (eds.), Hillsdale, NJ: Lawrence Erlbaum Associates, publishers, 1981.

Feldman, J.A., "Dynamic Connections in Neural Networks," *Biological Cybernetics*, 46, 27-39, 1982.

Feldman, J.A., "Four Frames Suffice: A Provisionary Model of Vision and Space," presented at the U. Mass. Workshop on Vision, Brain and Cooperative Computation, Amherst, Mass., May 1983; also, TR99, Computer Science Dept., University of Rochester, September 1982.

Feldman, J.A. and D.H. Ballard, "Connectionist Models and their Properties," *Cognitive Science*, 6, 205-254, 1982.

Hrechanyk, L.M. and D.H. Ballard, "Viewframes: A Connectionist Model of Form Perception," *Proceedings*, DARPA Workshop, June 1983.

Hrechanyk, L.M. and D.H. Ballard, "A Connectionist Model of Form Perception," *Proceedings*, Workshop on Computer Vision: Representation and Control, Rindge, New Hampshire, August 1982; also, *Computer Science and Engineering Research Review*, Computer Science Department, Fall, 1982.

Lampeter, W., "Design, Function and Performance of a System that Screens Chest Radiographs for Tumors," *Proceedings*, 1st CVPR Conference, June 1983.

Loui, R., "How Fast Hough?" Internal course project report, Computer Science Dept., University of Rochester, April 1983.

Rainero, E. and H. Kautz, "A Connection Chip," Internal course project report, Computer Science Dept., University of Rochester, April, 1983.

Russell, D.M., "Schema-based Problem Solving," forthcoming Ph.D. Dissertation, Computer Science Dept., University of Rochester, July 1983.

Sabbah, D., "A Connectionist Approach to Visual Recognition," TR107, Computer Science Dept., University of Rochester, April 1982; also Ph.D. thesis, Computer Science Dept., University of Rochester, April 1982.

Sher, D. and A. Tevanian, "A Hough Chip," Internal course project report, Computer Science Dept., University of Rochsster, April 1983.

Small, S.L., L. Shastri, M.L. Brucks, S.G. Kaufinan, G.W. Cottrell and S. Addanki, "ISCON: A Network Construction Aid and Simulator for Connectionist Models," TR109 Computer Science Dept., University of Rochester, September 1982.

Sloan, K.R., Jr. and L.M. Hrechanyk, "Surface Reconstruction from Sparse Data," *Proceedings: Pattern Recognition and Image Processing*, Dallas, Texas, August, 1981.

Stuth, B.H., D.H. Ballard and C.M. Brown, "Boundary Conditions in Multiple Intrinsic Images," *Proceedings*, IJCAI-83, Karlsruhe, West Germany, 1983.

IMAGE UNDERSTANDING RESEARCH AT USC:1982-83

R. Nevatia

Departments of Electrical Engineering
and Computer Science
University of Southern California
Los Angeles, California 90089-0272

I. INTRODUCTION

This paper summarizes our major research activities for the period of October 1982 to April 1983. More details can be found in other technical papers in the proceedings of this workshop [1-2] and the proceedings of the computer vision and pattern recognition conference being held concurrently with this workshop [3-5]. Our main focus has continued to be on developing a high-level symbolic matching system that would be useful for the tasks of map-updating, autonomous navigation and object recognition. We have largely used aerial images for testing, but the techniques should also apply to other domains. We have also been working on generating better descriptions, including improved segmentation, shadow analysis and stereo. We have also continued work with Hughes Research Laboratories on hardware implementation of IU algorithms.

II. SYMBOLIC MATCHING

Our recent work in this area has been primarily in extensive testing and evaluation of our previously reported matching methods [6-7]. We have compared our relaxation matching scheme to a variety of others, using different convergence and confidence updating criteria. These tests indicate that criterion optimization method is superior in terms of the number of iterations needed and in the accuracy of the results.

We have applied our line matching technique to the inspection of printed circuit boards for missing or improperly inserted parts. The system developed for aerial images required only small modifications to incorporate a more complex model representation; these efforts are described in [3].

III. STEREO MATCHING

Conventional stereo matching uses correlation of intensities in some form, in selected windows of two images. Some of the more modern approaches, e.g. [8,9], match edges, which are likely to be more invariant. We have recently started experimenting with matching of line segments. Initially, we attempted to apply our above cited line matching method; however, the distortions inherent in stereo images led us to a different matching criterion. Essentially, our system finds the set of matches that gives minimum "differential dispanity", i.e. the flattest consistent interpretation. This system needs further development, but the initial results are very promising; this work is described in detail in [1].

IV. SHADOW ANALYSIS

We have been working on using shadows to extract heights of structures; our work on extracting heights of buildings by using a priori knowledge of their shapes has been reported previously [10]. We have generalized this work for other objects, e.g. oil tanks, by using the known direction of illumination strongly to wake correspondence between object boundaries and their shadows. Some of our new work is described in [4].

V. SEGMENTATION

We have developed a new texture segmentation system that uses relatively simple measures of texture uniformity. The segmentation is hierarchical, a low resolution segmentation is used to compute a more accurate segmentation at a higher resolution level. The method is described in [2].

VI. REFERENCES

[1] G. Medioni and R. Nevatia, "Edge Based Stereo Matching," Proc. of DARPA Image Understanding Workshop, Arlington, VA, June 1982 (these proceedings).

[2] H.Y. Lee and K. Price, "Extraction of Textured Regions in Aerial Imagery," Proc. of DARPA Image Understanding Workshop, Arlington, VA, June 1983 (these proceedings).

[3] P. Kaufmann, G. Medioni and R. Nevatia, "Visual Inspection Using Linear Features," in Proceedings of Computer Vision and Pattern Recognition Conference, Washington D.C., June 1983.

[4] G. Medioni, ""Obtaining 3-D from Shadows in Aerial Images," in Proceedings of Computer Vision and Pattern Recognition Conference, Washington, D.C., June 1983.

[5] G. Medioni, "Matching Regions of Aerial Images," in Proceedings of Computer Vision and Pattern Recognition Conference, Washington, D.C. June 1983.

[6] O. Faugeras and K. Price, "Semantic Description of Aerial Images Using Stochastic Labeling," IEEE Trans-PAMI, Vol. 3, No. 6, November 1981, pp. 638-642.

[7] G. Medioni, "Matching Images and Maps," Proc. of the DARPA Image Understanding Workshop, Palo Alto, Ca., September 1982, pp. 103-111.

[8] W. Grimson, "From Images to Surfaces, MIT Press, Cambridge, MA 1981.

[9] H.H. Baker, "Depth from Edge and Intensity Based Stereo," Stanford Artificial Intelligence Laboratory Memo AIM 347, Stanford, CA, September 1982.

[10] A. Huertas, "An Edge Based System for Detecting Buildings in Aerial Images," in "Image Understanding Research," R. Nevatia, (Ed.), USC Report ISG 101, March 1982.

SECTION II

TECHNICAL PAPERS
PRESENTED

SURFACE CONSTRAINTS FROM LINEAR EXTENTS

John R. Kender
Department of Computer Science, Columbia University
New York, NY 10027

Abstract

This paper demonstrates how image features of linear extent (lengths and spacings) generate nearly image-independent constraints on underlying surface orientations. General constraints are derived from the shape-from-texture paradigm; then, certain special cases are shown to be especially useful. Under orthography, the assumption that two extents are equal is shown to be identical to the assumption that an image angle is a right angle (i.e. orthographic extent is a form of slope or skewed symmetry). Under perspective, if image extents are assumed equal and parallel, extent again degenerates into slope. In the general perspective case, the shape constraints are usually complex fourth-order equations, but they often simplify--even to graphic constructions in the image space itself. If image extents are colinear and assumed equal, the constraint equations reduce to second order, with several graphic analogs. If extents are adjacent as well, the equations are first order and the derived construction (the "jack-knife method") is particularly straightforward and general. This method works not only on measures of extent per texel, but also on reciprocal measures: texels per extent. Several examples and discussion indicate that the methods are robust, deriving surface information cheaply, without search, where other methods must fail.*

1 Introduction

In this paper, we show how certain simple aggregate image properties involving spatial extent along one dimension can be used as cues for determining underlying three-dimensional surface orientation. Image-measurable properties such as lengths and spacings are shown to generate constraints on local surface slope in a nearly image-independent way. The derivation of these relationships is identical in analytic method ("shape from texture") and representational structure (the gradient space) to those derived for other imaging phenomena such as skewed symmetry or image slope. Thus, they provide additional surface information in a form (either equation or graph) that is easily integrable with that of other existing algorithms.

Linear extents are measurements along a straight image line of either objects (in which case they are lengths) or virtual objects (in which case they are spacings). The exact form of the input to these analyses can vary. A prior edge-detection and linking step, or a segmentation-like step is assumed. Lengths are then linear measures of image tokens such as elongated blobs, and spacings are linear measures of the virtual lines between image tokens. Spacing behaves the same way as length does; often it is more conveniently available.

In general, this paper follows the image understanding conventions presented in, among other places, [Kender 80a]. That is, the image coordinate system considers the z axis to be positive in the direction of view; the image itself to be plane z=1, which has been rotated in front of the lens at the origin; and the unit of length in the system to equal the focal length of the lens. Surfaces in the scene are locally represented by planar patches, and the surface gradient of the patch z=px+qy+c is represented by the point (p,q), its gradient, in the gradient space.

The problem of deriving surface information from textural and regularity assumptions occurs in two steps. First, the textural element--in this case an image extent--is backprojected onto all surface patches possible. A map of the scenic measure of the component is recorded. The recovered scene extents are usually a function of the image extent's position and the surface's parameters. In the second step, two or more nearby textural elements are assumed to be equal in measure in the scene. Mathematically, this means that the maps can be intersected to find those surface patch parameters that generate for each texel the same measure (that is, the same texture). Because the gradient space is coupled to the image space--a rotation in one induces an equal rotation in the other--the problem of backprojecting textural elements often is can be simplified by factoring out rotation. That is, the camera roll component does not affect the depth and surface information of the image in any significant way.

2 Extents under Orthography

For the case of spatial extent under orthography, the rotational coupling reduces it to the problem of backprojecting a single horizontal extent between the points (a,y) and (b,y), where L=(b-a) is the image extent (see Figure 1). Further, the Jacobian of the deprojection mapping of image space onto the surface space is constant. (It is equal to ||N||, the norm of the surface normal N=(p,q,-1).) Thus all induced surface extents are preserved under translations of their sources in the image, and any candidate image extent can be translated to the origin. The problem then reduces to that of backprojecting the line from (0,0) to (L,0): a problem with one free parameter. Such simplifications characterize orthographic projection in general, but the resulting texture maps are often weak in analytie power, as the following discussion shows.

Backprojecting the image point (x,y,1) onto the plane with equation z=px+qy+c is achieved by the transformation: (x,y,1) becomes (x,y,px+qy+e). Without bothering to set up a detailed scene coordinate system, it is easy to see that the two points backproject to (0,0,e) and (L,0,pL+c), respectively. The scene extent is therefore L·sqrt($1+p^2$), which is a function of p and q: this is the normalized texture property map.

*This research was sponsored in part by the Defense Advanced Research Projects Agency under contract N00039-82-C-0427.

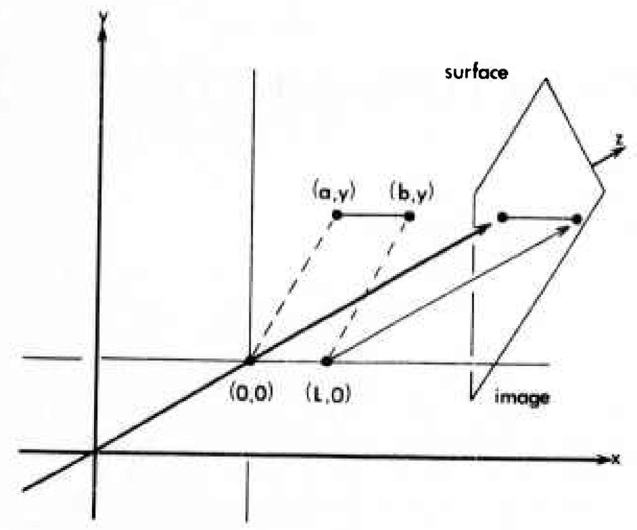


Figure 1: Backprojecting an extent under orthography.

Note that the q component does not affect the scene extent. This is because q measures departure from "verticality", and a horizontal line is only affected in backprojection by that component which alters "left-right" slant the p component). When graphed in the gradient space, this map has the property that the normalized texture property of extent is a hyperboloid of one sheet, with a minimum value of L on the q axis. Therefore, backprojection under orthography never decreases measures of extent.

To use this normalized texture property map (NTPM), consider an image with two extents in it. Suppose they arise from parallel extents in the scene; this parallelism is carried over into the image. But under orthography, either extent can be translated into superimposition on the other. Thus, if they are of equal extent, they (and their NTPMs) exactly coincide and no further information about the scene is obtained. If they are unequal in extent, they will superimpose with unequal overlap; since the coupling of the gradient space maps also causes the NTPMs to superimpose, there is no solution. That is, if L_1 and L_2 differ, L_1sqrt($1+p^2$) never equals L_2sqrt($1+p^2$), and there is no surface patch that can support two equal, parallel scene extents if there are unequal image extents. Thus parallelism of equal extents under orthography provides no information about surfaces, except in this weak, negative fashion.

Now suppose the extents arise from non-parallel, but equal extents in the scene. This situation is more interesting: the image extents can be translated so that a pair of their ends will meet and form an angle. Because the image extents are non-parallel in the image as well, their NTPMs have will have also been rotated different amounts. Further, their image measures are, in general, unequal, so the NTPM intersection is non-trivial. The resulting constraint equation is a messy one in terms of L_1, L_2, their joint angle, and second powers of p and q. However, it is not difficult to prove that the constraint on surface orientation that it induces can be graphed as a hyperbola in the gradient space. The following construction shows that the hyperbola is the Kanade hyperbola [Kender 80b], which usually arises under the assumption that a given image angle is caused by a scene right angle.

Consider Figure 2, in which the two image extents forming their angle have been closed off with the addition of a line. It is well known that orthography preserves midpoints of lines; thus the image figure, with yet another line connecting the vertex to this midpoint, can be seen as a scene isosceles triangle in perspective. Given this, the angle formed by the altitude to the base in the scene must be a right angle: this is the Kanade assumption. The surface constraint then is identically derived.

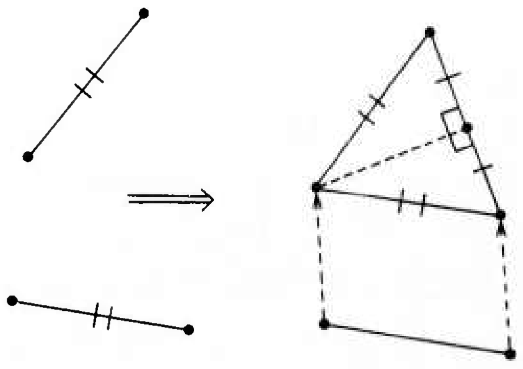


Figure 2: Equal extent is skewed symmetry.

A special case occurs when image extents are themselves equal, as with the corner of a square or rhombus. The altitude-to-base angle constructed in the image is found to be a true right angle as well. The second order constraint equation now degenerates to a linear one. Its graph in the gradient space is represented by two perpendicular lines through the origin; one of them is parallel to the triangle's base. These constrained surface orientations have a easy interpretation: the underlying surface could have pivoted about the altitude, foreshortening both halves of the triangle equally, or, the surface could have pivoted about the base, foreshortening the entire triangle, but without skew. Note that if the image is assumed to be that of a square corner, it can be analyzed solely in terms of slope phenomena. The scene then contains two right angles: the corner one and the induced one. (The two Kanade hyperbolae intersect in the gradient space, giving a Necker pair of orientations.)

Equal extents in the image under orthography therefore either give trivial results, or reduce to already known cases of image slope and angle. (This is even true for some other textural configurations for extents which are not covered here.)

3 Extents under Perspective

The analysis of extents under central perspective is more complex, but it yields more powerful algorithms for image understanding. Under perspective, the gradient space remains coupled to the image space, still saving one degree of freedom in analysis (the camera roll component). However, the backprojection function is more elaborate. In particular, the image point (x,y,1) is taken onto the surface $z=px+qy+c$ by $(-c/(1-px-qy))(x,y,1)$. This mapping has a non-linear Jacobian; therefore, the mapping of image extents into the scene extents is critically affected by translations in the image. This implies that the general backprojection of an image extent of measure L must be from (a,y,1) to (b,y,1), where $L=(b-a)$, since no simplifying translations are possible.

Thus, there are *three* free parameters. . The two points are taken into (-c/(1-pa-qy))(a,y,1) and (-c/(1-pb-qy))(a,y,1), respectively.

The induced surface extent is calculated in the scene by the usual Euclidean metric, yielding a complex NTPM:

$$L(1/(1-pa-qy))(1/(1-pb-qy))\mathrm{sqrt}((1-py)^2+p^2(y^2+1)).$$

(The calculation here, as in the orthographic case, is somewhat like a finite difference approximation to a derivative. However, under perspective, it is exactly the finite difference that is needed, since what is important is the departures from linearity.) For ease of reference, this NTPM will be abbreviated to

$$L(1/(1-pa-qy))(1/(1-pb-qy))S(p,y)$$

Notice that the function S(p,y) is independent of both a and b.

Theoretically (or even practically), this function is usable in its raw form. That is, given two extents in an image under central perspective, it is possible to generate the appropriate NTPMs for both (subject to their position and orientation), and to intersect their graphs--as if they were Hough accumulator arrays. The result would be a small set of surface orientations which would simultaneously normalize the two induced surface extents to equal measure. However, in nearly all cases, this involves the solutions to constraint equations that are of fourth order in p and q. Only a few image configurations generate simple surface constraints. (Some configurations that one might expect would reduce the complexity do not: for example, image extents that are radial with respect to the image origin). The ones that do simplify have the added benefit that they appear to be relatively common.

3.1 Equal and Parallel

First assume that image extents arose from scene components that were not only equal in measure, but were parallel on the scene surface. A simple construction (see Figure 3) shows that once again the image configuration can be handled solely by considerations of image slope. Two equal and parallel scene lines form a parallelogram; in the image, their pairs of sides can be extended to derive two vanishing points. Each vanishing point implies a linear constraint in the gradient space: if an image point (x,y) is a vanishing point of a surface, then the surface must have a gradient (p,q) which satisfies px+qy=1 ([Shafer 83]). Two such linear constraints uniquely define a vanishing line, which in turn uniquely defines the surface orientation.

3.2 Equal and Colinear

Assume now that the image extents did not arise from parallel scene extents. There seems to be only one other simplifying set of cases: those when the scene components are colinear. Although these cases also generate vanishing points, interestingly, they do not reduce the problem again to one of image slopes. Nothing can: colinear extents have only one slope in common.

The images of colinear scene components are also colinear. The reverse is not true, though the heuristic positing of that truth often is most useful. It would be yet another preference heuristic, similar to those used in other contexts in image understanding: for example, nearby image pixels arise from actual scene patch neighbors (shape from shading), nearly right angles arise from scene right angles (skewed symmetry), near-parallels arise from parallels (one form of shape from texture), etc.

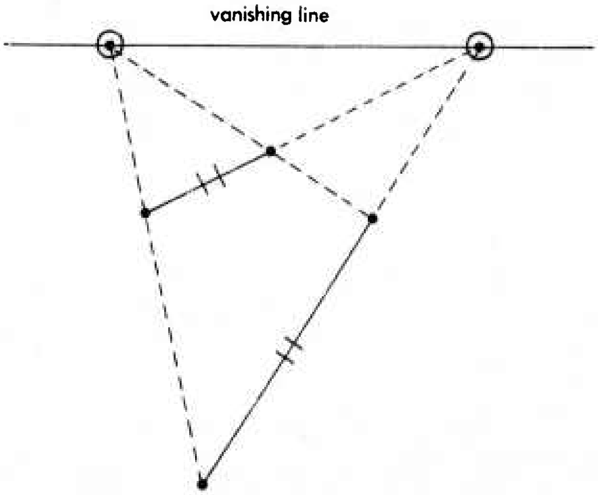


Figure 3: Equal parallels are equivalent to slope.

The image configuration in the most general case reduces to the following. Four points lie on the horizontal image line at height y; they are A=(a,y), with B, C, and D defined similarly). These four points define two image extents, L=(b-a) and R=(d-c), respectively. The assumption of colinearily allows the NTPMs of the extents to be put into correspondence easily: they are already in the proper orientation, due to the one shared image slope. Since they also share identical terms in S(p,y), equating the NTPM yields a surface constraint that reduces to second order in p and q:

$$(1-pa-qy)(1-pb-qy)/L = (1-pc-qy)(1-pd-qy)/R$$

Although this equation can be exactly solved, it has a simplifying graphic construction that can be drawn in the image space itself, directly yielding the vanishing point(s). Rewrite it in the following form:

$$(X-a)(X-b)/L = (X-c)(X-d)/R, \text{ where } X = (1-qy)/p$$

If X satisfies the constraint equation, then scene extents are equal, as desired. Further, this is a very desirable X: it also satisfies the formal definition that pX+qy=1, that is, the point (X,y) is a vanishing point. Note that (X,y) lies on the line of colinearity; all that must be calculated is the value of X itself. Formally, the equation is of the form of the intersection of two parabolae. The left parabola has value 0 at both a and b, and a minimum value of L/4 midway between them. The right parabola is exactly of the same shape, except for scaling (its midpoint minimum is R/4). Thus, the value of X can be graphically determined by drawing the parabolae on the image, and finding their intersection. (Notice that the mathematics, as well as the construction, finds a vanishing point between b and c, where the image lengths are on *opposite* sides of any vanishing line.)

The parabola method can be refined in the following way. Note first that it is not necessary to draw the parabolae on the x axis; they can be translated upwards to the horizontal line of colinearity itself. Secondly, the parabola are only constrained to pass through the point pairs; their exact shape is not critical, as long as the parabolae are similar (i.e. they can be mutually scaled). Third, since the value of X is a purely formal one, the parabolae can be imagined to be drawn *out of* the image plane: that is, either parabola can be though of as extending into the -z axis direction.

More appropriately, the value of X on either parabola can be considered as an image feature in its own right. The calculation is really a type of local feature assignment, with with each position on the line of colinearity being assigned two simultaneous features. That position where the features are identical is the vanishing point.

Parabolae grow very quickly, however, away from their roots. This can be compensated for formally by taking the square root of this image feature. The assignment of values is now via hyperbolae of similar shape, which grow (sub)linearly. They also have the (aesthetic) advantage of being undefined within the image extents themselves, the interior of which being one place where a vanishing point ought not be. In a pinch, the hyperbolae can also be approximated by their asymptotes, which, being strictly linear, are easier to compute. For example, the left hyperbola is sqrt((X-a)(X-b)/L); its asymptotes originate at the left texel's midpoint, and have slopes of sqrt(L) and -sqrt(L) (see Figure 4). Still other modifications and approximations of this formal equation are possible; they would need to be analyzed for accuracy and computational efficiency.

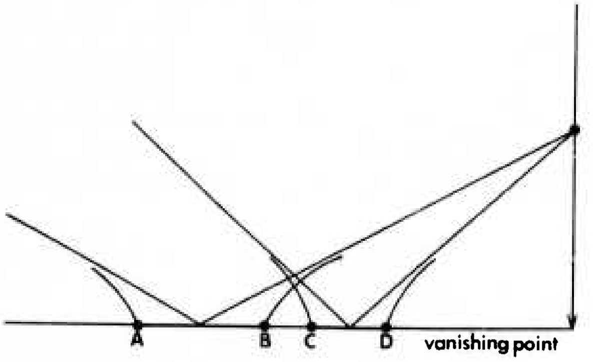


Figure 4: The hyperbola and asymptote methods.

3.3 Equal, Colinear, and Adjacent

The last special case is the simplest, but perhaps the most powerful. Suppose that two colinear and *adjacent* image extents are derived from two colinear, adjacent, and equal scene components. That is, as in Figure 5, the points B and C have merged. Then the constraint given for the general four-point colinear case simplies even further since B=C, to that of a linear constraint in p and q:

$$(1\text{-}pa\text{-}qy)/L = (1\text{-}pd\text{-}qy)/R$$

By the same formal method as above, it can be rewritten as:

$$(X\text{-}a)/L = (X\text{-}d)/R, \text{ where } X = (1\text{-}qy)/p$$

Either side is the equation of a line. With exactly the same flexibilities of the parabola scheme above, these lines can be plotted in the image space (see Figure 5). That is, they can extend out of the image in the -z direction; they can be mutually scaled; X can again be considered an image feature, labeling each position on the line of colinearity with a two-tuple of features. As before, the vanishing point occurs when the features are equal; this occurs at X=(Ld-Ra)/(L-R).

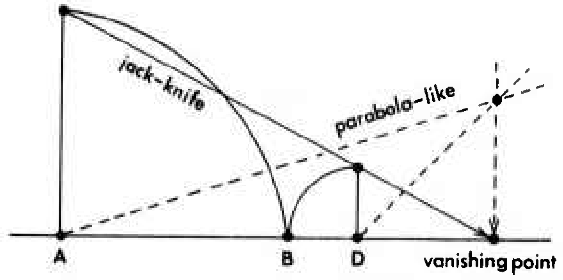


Figure 5: "Jack-knife" method for vanishing points.

Yet another graphic construction is possible. It too has a feature space interpretation, this time very useful. Construct at A a feature of value L; conceptually, this is constructed by a line of length L perpendicular to the line of colinearity. (Alternatively, the line can point in the -z direction.) Similarly construct at D a feature of value R. The resulting figure may resemble a jack-knife, with its two blades opened in parallel, outwards. (As with a jack-knife, the blades do not need to be perpendicular to their base; however, for the method to work, the blades must be parallel. The proof is by similar triangles.) Then under this interpretation, the feature values of all other points on the line of colinearity are determined by linear extrapolation from the two given ones. That is, values are generated from this new X by (R(X-a)-L(X-d))/(L+R). In particular, the vanishing point is where this image feature value is 0, as can be verified by direct substitution. It is not hard to show that this construction really does implement an image feature: it is scaled inverse depth.

These methods are formal; as with the parabola method, other modifications of the constraint equation are possible as well. It should be noted that the jack-knife equation can also be derived from the application of methods of projective geometry: either through the cross-ratio, or through the appropriate nine-point geometric construction. The parabola method apparently cannot, however, as it deals with five points at a time.

3.4 A Reciprocal Method

The jack-knife method has an interesting extension. The primary heuristic assumption required for its use only requires that image extents arise from equal surface extents; however, what is meant by extent can be defined in many ways. In particular, a series of N extents laid colinearly end to end on a surface can be considered either as a one extent of length N, or N of length one (or many other combinations). Often, runs of multiple extents can be obtained by looking for repeated distinguishing events along an arbitrary line through the image. (Strong edges of the same polarity, say, are events: see Figure 6). The prior jack-knife method would try to normalize the extent of the entire run. But under the assumption that the events form a texture, the method can be extended to normalize each event as well.

Computing Visual Correspondence

Michael Kass
MIT Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

Abstract

A computational framework for solving the visual correspondence problem is presented and evaluated by using a stochastic image model. The framework differs from previous work in that it emphasizes the combination of a large collection of independent measurements. Partial derivatives of images smoothed with a few different-sized Gaussian filters are suggested as suitable measurements. A specific computation is shown based on a stochastic image model to reliably establish whether or not two points correspond, provided that the signal to correspondence noise ratio in the images to be matched exceeds two. The computation has been applied to artificial and natural images with encouraging results.

1. Introduction

The problem of matching up two similar views of the same scene is one of the critical problems which any powerful vision system must solve. Known as the correspondence problem, it occurs most notably in stereopsis where the two views come from separate vantage points, and in motion analysis where the two views come from the same vantage point but are separated in time. In stereopsis, a solution to the correspondence problem yields relative depth information, while in motion analysis, it yields information which can be used to segment an image into regions belonging to different objects and can be used to approximate their velocities. The human visual system is known to solve both correspondence problems with impressive range, resolution, and noise immunity though the manner in which it does so is ill understood. Computer solutions to the correspondence problem have fallen far short of similar performance, particularly on natural images. A new approach to the problem will be presented here which, it is hoped, will provide insight into the structure of the correspondence problem, as well as a robust technique for solving it.

The correspondence problem can be stated quite simply as follows. Given two similar images of the same scene, a point in the first image is said to correspond with a point in the second image if both are projections along lines of sight of the same physical point. The correspondence problem consists of trying to match up as many pairs of corresponding points as possible given the intensity profiles of the two similar images.

All algorithmic solutions to the problem are based on the idea that the light intensity profiles surrounding corresponding points are quite similar. For each point in the first image, only points in the second image with quite similar local intensity profiles need be considered as potential matches. If the similarity measure is chosen appropriately, then a large fraction of the points in the first image will have only one potential match in the second. If it can be confidently determined that the similarity between these points and their potential matches is not due to chance, then the unique potential matches can be trusted as correct matches. Global consistency constraints can be used in some cases to choose among several potential matches, but this may not be necessary if the local information is extracted properly.

Choosing a good measure of similarity for the correspondence problem is quite difficult. Corresponding points often have substantially different light intensity values because of the different viewing angles. More importantly, at depth discontinuities in stereopsis or object boundaries in motion analysis, the light intensity values surrounding two corresponding points can be quite different. When specular reflection and assorted sources of noise are also considered, it becomes clear that the similarity measure should be chosen quite carefully.

Two classes of similarity measures have been investigated in the literature. The first consists of traditional statistical measures such as correlation and mean square error (e.g. [Gennery 77], [Moravec 77]). While algorithms based on these measures have seen some success, their performance has been rather disappointing as a whole. Except under controlled conditions, the intensity profiles of corresponding points are usually not correlated enough

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's Artificial Intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-75-C-0643, the Office of Naval Research under contract number N0014-80-C-0505, and the System Development Foundation.

It does this by simply dividing normalized run extent by event count. Thus, given a ru i of events, the extended method divides it into two sections, each with an image extent and an event count, and solves the modified equation:

$l(1\text{-}pa\text{-}qy)/L=r(1\text{-}pd\text{-}qy)/R$, where l and r are event counts

Note that the run can be split in many places, and that the modified equation can be solved by any of the techniqu s given in the jack-knife method (with L and R appropriately modified to L/l and R/r, respectively.) The optimal ways to split the run would have to be analyzed.

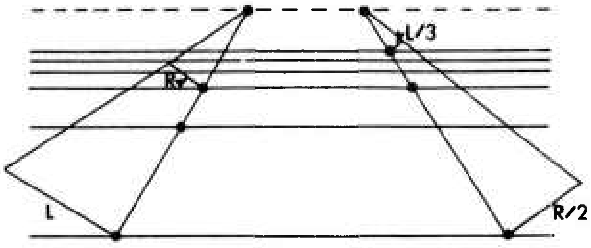


Figure 6: Jack-knife methods on a wave-like texture.

The jack-knife method is based on a measure of extent-per-texel; this reciprocal method uses texels-per-extent. The reciprocal method has many advantages. The two sections that count events can be of fixed image size and location. Within each section, event counts can be recovered by simple pattern recognition techniques. The final computation is simple. In effect, the shape constraints under this method come from simple feature detectors.

3.5 Examples and Comment

The true beauty of the jack-knife methods comes from the fact that they are one-step and robust.

At least two other methods for determining surface orientation rely on an implicit searching for image "regularity"; having found it they postulate the vanishing line to be parallel to it. The remaining surface constraint is determined by different means [Bajcsy 76; Stevens 79]. Here, the two steps are integrated; "tilt" need not be found before "slant", since any two vanishing points will do.

The jack-knife methods succeed even with difficult textures or orientations. As in the wave texture of Figure 6, sometimes the vanishing line direction has *no* measurable regularity; regularity-based tilt-searches must fail. The jack-knife methods will return a proper vanishi.g point, however, as long as they are *not* aligned with the vanishing line. The jack-knife methods even work without search on frontal ((p,q)=(0,0)) textures, in which *every* direction exhibits image textural regularity. In this case, the jack-knife methods properly return infinite vanishing points.

Acknowledgements

I thank Kerny Calaway for her graphic skills.

References

[Bajcsy 76] Bajcsy, R., and Lieberman, L. "Texture Gradient as a Depth Cue." *Computer Graphics and Image Processing 5*, 5 (March 1976), 52-67.

[Kender 80a] Kender, J.R. *Shape from Texture.* Ph.D. Thesis, Carnegie-Mellon University Computer Science Department, Nov. 1980.

[Kender 80b] Kender, J.R., and Kanade, T. Mapping Image Properties into Shape Constraints: Skewed Symmetry, Affine-Transformable Patterns, and the Shape-from-Texture Paradigm. Proceedings of the First Annual National Conference on Artificial Intelligence, American Association for Artificial Intelligence, Aug., 1980, pp. 4-6.

[Shafer 83] Shafer, S.A., Kanade, T., and Kender, J.R. "Gradient Space under Orthography and Perspective." *Computer Graphics and Image Processing* (To appear 1983).

[Stevens 79] Stevens, K.A. *Surface Perception from Local Analysis of Texture and Contour.* Ph.D. Thesis, MIT Artificial Intelligence Lab., Feb. 1979. Available as AI-TR-512

for these algorithms to work reliably on very many points in an image. Difficulties with this class of similarity measures have led a number of researchers to examine a second class of similarity measure, those based on edge finding (e.g. [Marr and Poggio 79], [Grimson 81] [Baker and Binford 81]). These measures assert that two points are similar if and only if they both lie on edges of approximately the same orientation. More encouraging results have been obtained with these methods, but important problems remain. Perhaps chief among these is the problem of occlusion. Physical points visible in only one of the images tend to get matched spuriously by edge based algorithms. Chance matches for these points can frequently be found and are difficult to prune. Since such points occur principally at object boundaries, they are arguably the most important points to deal with effectively.

This paper will describe a third kind of similarity measure for the correspondence problem. Based on the idea of combining independent measurements, the measure has remarkable noise immunity and works reliably at occluding contours. No single image measurement in this approach is trusted to indicate very much about the correspondence of a pair of images. Unanimity among the independent measurements, however, is taken as a powerful indication of correspondence. Because information from a large number of measurements is combined, the approach is far more robust in a number of important ways, than approaches which rely heavily on a very small number of measurements. As a consequence, the solution to be presented here can be expected to work quite well in a wide variety of viewing conditions.

2. Similarity Measurement

Let $I_1(x,y)$ and $I_2(x,y)$ be the light intensity functions for two images whose correspondence is to be computed and let $D(x,y)$ be the true offset or disparity between the images measured relative to the coordinate system of $I_1(x,y)$ and defined on some set of points $\mathcal{D} \subseteq \Re^2$ for which corresponding physical points are visible in both images. Then for all $p \in \mathcal{D}$, $I_1(p)$ and $I_2(p + D(p))$ are projections of the same physical point. The problem is to recover D from I_1 and I_2.

Measuring similarity can be thought of as a two step process. The first step is to create a representation of the local intensity variation at every point in each of the two images. The second step is to compare the local representations and determine how close they are to each other. In the general case, the representation consists of a collection $f_i(p,I)$, $1 < i < n$ of different image functionals (filters). For edge-based approaches, the functionals would measure the presence or absence of different classes of edges, and for correlation approaches they would measure weighted local image intensities.

In order to make use of the full power of statistical combination, we need the functionals to be both numerous and nearly independent. Typical sets of edge based functionals are insufficient in number, and correlation based functionals are not independent, so neither set is appropriate for reliable statistical inference. The typical edge-based functionals could be supplemented by others, but that will not be investigated here. Instead, the simplest interesting class of functionals — linear ones — will be considered. One reasonable set of nearly independent linear functionals will be presented in section 5. For the moment, assume such a set exists.

Each functional in the local intensity representation implicitly defines a similarity measure for correspondence since we expect that $f_i(p,I_1) \approx f_i(p + D(p), I_2)$ provided the f_i are chosen carefully. If we combine the functionals into a vector at each point: $F(p,I) = (f_1(p,I), f_2(p,I), \ldots, f_n(p,I))$ then we can expect the vector $F(p_1,I_1) - F(p_2,I_2)$ to be very small in each component if p_1 and p_2 correspond. On the other hand, if p_1 and p_2 do not correspond, it is likely that $F(p_1,I_1) - F(p_2,I_2)$ has at least one large component.

The above intuition can be translated into an algorithm as follows. Define $matchp_i(p_1,p_2)$ be a predicate which is true if and only if

$$|f_i(p_1,I_1) - f_i(p_2,I_2)| < k_i\sigma(f_i(p,I_1))$$

where $\sigma(x)$ denotes the square root of the expected value of x^2 and let $matchp(p_1,p_2)$ be a predicate which is true if and only if for all $i \in \{1,2,\ldots,n\}$,$matchp_i(p_1,p_2)$. Then $matchp$ is true of a pair of points p_1 and p_2 if each component of $F(p_1,I_1) - F(p_2,I_2)$ is smaller than its globally determined threshold. It will be argued that $matchp$ does a good job of solving the correspondence problem if the f_i and the k_i are chosen appropriately—it is almost always true of corresponding points and almost never true of non-corresponding points.

3. Expected Error Rates

In order to evaluate $matchp$, suppose the f_i are orthogonal linear shift invariant functionals and consider the following stationary image model. Let I_1 be stationary Gaussian white noise and let I_2 be derived from I_1 by shifting it according to $D(p)$ and adding Gaussian white correspondence noise, $N(p)$. The efficacy with which $D(x,y)$ can be determined from the f_i under these conditions depends upon how well the f_i are preserved between views. Let

$$SNR_i = \sigma(f_i(p,I_1))/\sigma(f_i(p,N))$$

be the signal-to-correspondence-noise ratio of the ith functional. If SNR_i is greater than two for a dozen functionals, then $matchp$ will very reliably determine whether or not two points correspond.

Three performance criteria will be considered for *matchp*. The first is the rate of false positives—the probability that *matchp* will be true of two non-corresponding points. The second criterion is the rate of false negatives–the probability that *matchp* will be false of two corresponding points. The third criterion is one of resolution and concerns the extent to which corresponding points can be spatially localized.

The calculation of the false negative rate is relatively straightforward. Let p_1 be a randomly selected point in $\mathcal{D}$. The difference between the ith functional evaluated at p_1 and the same functional evaluated at the corresponding point $C(p_1)$ is equal to the value of the functional applied to the correspondence noise. A false negative occurs when that difference exceeds the threshold. The distribution of $f_i(p, N)$ is normal since it is a convolution with a Gaussian process. The probability that it exceeds the threshold is the false negative rate for $matchp_i$ and is given by

$$Pr[\sim matchp_i(p_1, C(p_1))] = 1 - \text{erf}\left(\frac{\sqrt{2}}{2} k_i SNR_i\right).$$

A false negative occurs for *matchp* when a false negative occurs for any of the $matchp_i$ predicates. Since the functionals are independent, the false negative rate for *matchp* is

$$Pr[\sim matchp(p_1, C(p_1))] = 1 - \prod_{i=1}^{n} \text{erf}\left(\frac{\sqrt{2}}{2} k_i SNR_i\right).$$

The false positive rate is also easy to calculate. Let p_1 and p_2 be two randomly selected non-corresponding points. The difference between $f_i(p_1, I_1)$ and $f_i(p_2, I_2)$ is a normally distributed random variable with standard deviation

$$\begin{aligned}\sigma(f_i(p_1, I_1) - f_i(p_2, I_2)) &= \sqrt{\sigma^2(f_i(p, I_1)) + \sigma^2(f_i(p, I_2))} \\ &\approx \sqrt{2}\sigma(f_i(p, I_1))\end{aligned}$$

where the approximation is based on the assumption that $\sigma(f_i(p, I_1)) \approx \sigma(f_i(p, I_2))$. Thus the probability of a false positive based on the ith functional is just the probability that a normal random variable with the above standard deviation has magnitude below the threshold. That probability is given by

$$Pr[matchp_i(p_1, p_2)] \approx \text{erf}\left(\frac{k_i}{2}\right).$$

A false positive for *matchp* occurs only when a false positive occurs for each of the $matchp_i$ predicates. Hence the probability of a false positive is just

$$Pr[matchp(p_1, p_2)] \approx \prod_{i=1}^{n} \text{erf}\left(\frac{k_i}{2}\right)$$

due to the independence of the functionals.

Suppose $SNR_i = 2$ for all i and $n = 12$. Then the choice of k_i represents a tradeoff between a very low false positive rate and a very low false negative rate. False positives often result in the generation of wrong disparity values so they tend to be quite serious. False negatives, on the other hand, usually result simply in not being able to determine the disparity at a particular point. Thus a reasonable choice of k_i is one which produces a negligable false positive rate while still keeping the false negative rate to a low level. One such choice is $k_i = 1.2$. The resulting false positive rate is .2 per cent and the resulting false negative rate is 18 per cent. Both rates are a good deal lower than what is needed to reliably determine image correspondence. If the signal to correspondence noise ratio is improved to three, the false positive rate can be improved an order of magnitude without worsening the false negative rate.

4. Expected Resolution

The third criterion of performance for *matchp* is that of resolution. If p_1 is picked at random and $p_2 = C(p_1) + r$ then if r is small enough, $Pr[matchp(p_1, p_2)]$ will be quite large. The separation r at which $Pr[matchp(p_1, p_2)]$ becomes small will determine the resolution with which disparity can be recovered using *matchp*. Let A_i be the autocorrelation function for $f_i(p, I_2)$ on I_2 defined as

$$A_i(x, y) = \frac{f_i(p, I_2) * f_i(p, I_2)}{\sigma^2(f_i(p, I_2))}$$

where the asterisk denotes convolution. Then $f_i(C(p_1), I_2)$ and $f_i(C(p_1) + r, I_2)$ have a joint normal distribution with correlation $A_i(r)$. The density of the distribution is

$$\frac{1}{2\pi}(\det \Sigma)^{-1/2} e^{-X^T \Sigma^{-1} X/2}$$

where X is the vector $(f_i(C(p_1), I_2), f_i(C(p_1) + r, I_2))$, X^T is X transpose and Σ is the covariance matrix:

$$\sigma^2(f_i(p, I_2)) \begin{pmatrix} 1 & A_i(r) \\ A_i(r) & 1 \end{pmatrix}$$

Consider first the case where the correspondence noise is zero. Then we are interested in the probability

$$v_i(r) = Pr[|f_i(C(p_1), I_2) - f_i(C(p_1) + r, I_2)| < k_i \sigma(f_i(p, I_2))]$$

that one of the functionals evaluated at two points separated by r does not change enough to produce differing values of *matchp*$_i$ at those points. Integrating the joint normal density over the area where $|f_i(C(p_1), I_2) - f_i(C(p_1) + r, I_2)| < k_i\sigma(f_i(p, I_2))$ yields

$$v_i(r) = \mathrm{erf}\left(\frac{k_i}{2\sqrt{1 - A_i(r)}}\right)$$

For a functional whose impulse response has finite energy, $A_i(r)$ must asymptotically approach zero as r becomes large. As a consequenee,

$$\lim_{|r|\to\infty} v_i(r) = \mathrm{erf}\left(\frac{k_i}{2}\right).$$

This should come as no surprise since the left side is the probability that two points separated by r differ in the ith functional by more than the threshold which should be equal to the false positive rate for *matchp*$_i$ in the limit as $|r| \to \infty$.

Now consider the impact of correspondence noise on the resolution. The probability of $matchp_i(p_1, C(p_1) + r)$ being true is equal to the probability that $f_i(C(p_1) + r, I_2)$ falls in the interval $B = [f_i(p_1, I_1) - m_i, f_i(p_1, I_1) + m_i]$ where $m_i = k_i\sigma(f_i(p_1, I_1))$ is the threshold for *matchp*$_i$. Suppose $matchp_i(p_1, C(p_1))$ is true. Then $f_i(C(p_1, I_2))$ is by definition contained in the interval B. Since B has length $2m_i$, it must be contained in the interval $C = [f_i(C(p_1), I_2) - 2m_i, f_i(C(p_1), I_2) + 2m_i]$. Thus the probability that $matchp_i(p_1, C(p_1) + r$ is true is less than the probability that $f_i(C(p_1) + r, I_2)$ falls in the interval C, a probability that can be calculated as before to be

$$Pr[f_i(C(p_1) + r, I_2) \in C] = \mathrm{erf}\left(\frac{k_i}{\sqrt{1 - A_i(r)}}\right).$$

Hence, in the presence of correspondence noise, the resolution of *matchp*$_i$ with $k_i = k$ for points which it correctly matches can be no worse than the resolution of *matchp*$_i$ in the absence of noise with $k_i = 2k$. Let $v_i^*(r)$ be the probability that $matchp(p_1, C(p_1) + r)$ is true given that $matchp(p_1, C(p_1))$ is true. Then

$$v_i^*(r) > \mathrm{erf}\left(\frac{k_i}{\sqrt{1 - A_i(r)}}\right).$$

If any of the *matchp*$_i$ can resolve the disparity to within r, then *matchp* will also be able to do so. A conservative estimate of its resolution is expressed by the relation

$$Pr[matchp(p_1, C(p_1) + r) | matchp(p_1, C(p_1))] =$$

$$v^*(r) > \prod_{i=1}^{n} \mathrm{erf}\left(\frac{k_i}{2\sqrt{1 - A_i(r)}}\right)$$

Functionals whose autocorrelation function fall off slowly with distance from the origin will not affect the resolution very much since their contribution to the above probability will be multiplication by a factor near one in the area of interest. On the other hand, a functional with a sharply peaked autocorrelation function will strongly affect the resolution.

One useful measure of resolution is the distance r_s at which the probability of discrimination drops to fifty per cent. A very conservative estimate of r_s can be produced by looking only at the functional with the most strongly peaked autocorrelation function and using the conservative estimate developed above for the resolution of a single functional in the presence of correspondence noise. Suppose

$$.5 = \mathrm{erf}\left(\frac{k_i}{\sqrt{1 - A_i(r)}}\right).$$

Then

$$\frac{k_i}{\sqrt{1 - A_i(r)}} \approx \frac{2}{3}.$$

Using a second order Taylor expansion for $A_i(r)$, we obtain

$$r_s < r \approx \frac{3}{2}k_i\left(\frac{\partial^2 A_i}{\partial r^2}\right)^{-1/2}\Bigg|_0$$

Thus the separation at which fifty per cent discrimination occurs using *matchp*$_i$ is approximately proportional to the threshold k_i and inversely proportional to the square root of the curvature of the autocorrelation of the ith functional at zero. The resolution of *matchp* can be expected to be a good deal better than the best resolution of the *matchp*$_i$.

5. Choice of Measurements

In deriving the properties of *matchp* which allow it to be used to solve the correspondence problem, the existence of a set of a set of independent, linear shift invariant functionals whose values are loosely preserved between views was assumed. One such set will be presented here. If p_1 is a point in D and p_2 is its corresponding point then the functions $I_1(p_1 + r)$ and $I_2(C(p_1) + r)$ can be expected to be quite similar for small values of r. One complete characterization of the local behavior of a function of two dimensions is its two dimensional Taylor series, so it is natural to examine derivatives of $I_1(p_1 + r)$ and $I_2(C(p_1) + r)$. As one might expect, first and second partial derivatives appear empirically to be fairly well preserved between views. Differentiation tends to accentuate noise, however, so it is usually a good idea to do some low pass filtering before taking any sort of derivative. Marr and Hildreth [1980] argue that the best low pass filter to use for applications such as this is a filter with a Gaussian impulse response because it minimizes the product of localization in space and frequency. Hence a reasonable set of functionals to look at is the set of derivatives of Gaussian smoothed images.

Not all derivatives of Gaussian smoothed images are independent. In fact, the nth and $n+2$nd derivatives of Gaussian smoothed white noise are very strongly correlated. These correlations can be calculated fairly directly.

The Gaussian mask normalized to have unit integral is

$$f_\sigma(x,y) = \frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$$

For notational convenience, define

$$f_{n,m,\sigma}(x,y) = \frac{\partial^n}{\partial x^n}\frac{\partial^m}{\partial y^m}f_\sigma(x,y)$$

The desired correlation is

$$Corr(f_{n_1,m_1,\sigma_1}, f_{n_2,m_2,\sigma_2}) = \frac{\int\int f_{n_1,m_1,\sigma_1} f_{n_2,m_2,\sigma_2}dxdy}{\sqrt{(\int\int f^2_{n_1,m_1,\sigma_1}dxdy)(\int\int f^2_{n_2,m_2,\sigma_2}dxdy)}}$$

where the integration goes from negative infinity to positive infinity.

Straightforward calculations [Kass 82] show that the magnitude of the correlation is just

$$Corr(f_{n_1,m_1,\sigma_1}, f_{n_2,m_2,\sigma_2}) = \left(\frac{2\sigma_1\sigma_2}{\sigma_1^2+\sigma_2^2}\right)^{m+n+2}\frac{n!m!}{(n/2)!(m/2)!}\sqrt{\frac{n_1!m_1!n_2!m_2!}{(2n_1)!(2m_1)!(2n_2)!(2m_2)!}}$$

The following table gives the correlations for the case where $m_1 = m_2 = 0$ and $\sigma_1 = \sigma_2$. Note the high correlation between $f_{n,0,\sigma}$ and $f_{n+2,0,\sigma}$.

	f	$\partial f/\partial x$	$\partial^2 f/\partial x^2$	$\partial^3 f/\partial x^3$	$\partial^4 f/\partial x^4$
f	1.	0.	-.58	0.	.29
$\partial f/\partial x$	0.	1.	0.	-.77	0.
$\partial^2 f/\partial x^2$	-.58	0.	1.	0.	-.85
$\partial^3 f/\partial x^3$	0.	-.77	0.	1.	0.
$\partial^4 f/\partial x^4$	.29	0.	-.85	0.	1.

The high correlations in the above table suggest that most of the usable information in the local behavior of an image at a point p is contained in a maximal set of independent terms of the Taylor series around p. There is a high correlation between f_{n_1,m_1,σ_1} and f_{n_2,m_2,σ_2} when $n_1 \equiv n_2$ (mod 2) and $m_1 \equiv m_2$ (mod 2). Hence no independent set of Taylor series terms for images can have more than four elements, one for each possible combination of n (mod 2) and m (mod 2) where n and m are the number of derivatives taken in the x and y directions. One maximal set of independent functionals is given by the following set of derivatives of Guassian smoothed white noise.

$$\mathcal{F}_\sigma = \left\{\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial^2 f}{\partial x^2}, \frac{\partial^2 f}{\partial y^2}\right\}$$

A larger set of approximately independent functionals can be constructed by considering different amounts of Gaussian smoothing. If the ratio between the standard deviations of the two Gaussians is $s = \sigma_1/\sigma_2$ then effect of the size of the Gaussians on their correlation can be expressed by the relation

$$Corr(f_{n_1,m_1,\sigma_1}, f_{n_2,m_2,\sigma_2}) = \left(\frac{2s}{s^2+1}\right)^{m+n+2} Corr(f_{n_1,m_1,\sigma_1}, f_{n_1,m_1,\sigma_1})$$

The maximum correlation between two functionals in $\mathcal{F}_{\sigma_1} \cup \mathcal{F}_{\sigma_2}$ is therefore $(2s/(s^2+1))^4$ which occurs between first order terms. If $s = 2$, the correlation is .41 but if $s = 2.5$, it drops to .23 and if $s = 3$, it falls to .13. The impact of these small, non-zero correlations on the performance of *matchp* can safely be ignored. If the number of different Gaussians is increased to three, the largest correlation does not increase. Thus $\mathcal{F}^* = \mathcal{F}_\sigma \cup \mathcal{F}_{\sigma s} \cup \mathcal{F}_{\sigma s^2}$ defines a a set of twelve functionals in which the largest pairwise correlation is still $(2s/(s^2+1))^4$. If s is at least 2.5, then the twelve functionals in $\mathcal{F}^*$ will have sufficiently low correlations to be regarded as approximately independent. Since they are all linear and shift-invariant, they will satisfy all the conditions on the f_i used in deriving the performance of *matchp*.

A conservative estimate of the probability that a particular functional will be unable to resolve the disparity of a point to better than an uncertainty of r was previously calculated in terms of the most sharply peaked autocorrelation. The autocorrelation function of $f_{n,m,\sigma}$ is just

$$f_{n,m,\sigma} * f_{n,m,\sigma} = f_{2n,2m,\sqrt{2}\sigma}$$

so the probability that the functional with impulse response

$f_{n,m,\sigma}$ will be unable to localize the disparity of a pair of images to within a range smaller than r is no larger than

$$v_i(r) = \operatorname{erf}\left(\frac{k_i}{\sqrt{1 - f_{2n,2m,\sqrt{2}\sigma}(r)}}\right).$$

The best resolution along the x-axis for functionals in $\mathcal{F}$ occurs with the functional that has an impulse response equal to $f_{2,0}$. Its autocorrelation function is

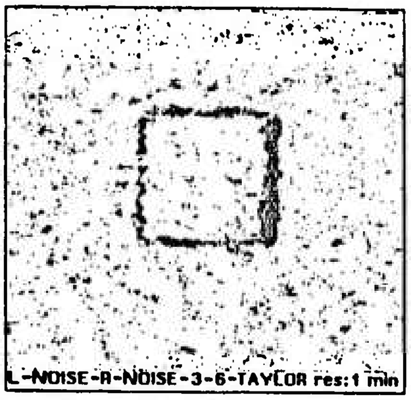
$$\begin{aligned} f_{2,0,\sigma} * f_{2,0,\sigma} &= f_{4,0,\sqrt{2}\sigma} \\ &= \left(\frac{x^4 - 48x^2\sigma^4 + 48\sigma^2}{64\pi\sigma^{10}}\right)e^{-(x^2+y^2)/4\sigma^2} \end{aligned}$$

The probability that none of the functionals will be able to resolve the disparity to within r is the product of the v_i and measures the resolution of *matchp.*

6. Empirical Performance

As an initial test, *matchp* was applied to a pair of gray level images generated by computer with all the important characteristics of the image model used here.

Figure 1. ***Matchp*** **applied to a Julesz random dot stereogram. Dark points were unmatched**



$\mathcal{F}^*$ was used as the set of functionals. The image pair has a signal-to-correspondence-noise ratio of two and a disparity field which is zero everywhere except in a central square covering one ninth of the image area where it is $(6,0)$ in pixels. For each point (x,y) in the left image, $matchp(x+n,y), n \in \{-8,-7,-6,\ldots,6,7,8\}$ was calculated. If there was only one n such that $matchp(x+n,y)$ was true, n was recorded as the disparity value. If there was more than one n such that $matchp(x+n,y)$ was true, the disparity was recorded as ambiguous. If there was no n such that $matchp(x+n,y)$ was true, the disparity was recorded as unknown.

Figure one shows the results of applying *matchp* to the above pair of images. The dark points indicate areas where the algorithm was unable to find matches. Slightly over 93 percent of the pixels in each image were uniquely matched. The mean square error in the disparity values generated was a small fraction of a pixel despite the large amount of correspondence noise.

It is worth noting that *matchp* failed to match most of the points along the border of the shifted central square. Some of the points were occluded in the second image, so it was correct not to match them, but most of the points went unmatched because the steep disparity gradient on the border substantially decreased the signal-to-noise ratio. Correlation and edge-based algorithms tend to generate significant numbers of incorrect disparity values at occluded regions and at places where the disparity gradient is large, but the algorithm based on *matchp* avoids doing so because of *matchp*'s unusually low false positive rate.

Matchp has been applied to a small number of natural images as well. Figure 3 shows the results of interpolating a surface through disparity values generated by *matchp* for the stereo pair in figure 2. Intensity is proportional to depth. The photos are of the campus of the University of British Columbia and were obtained from the B.C. Ministry of Forests.

The combination of independent results has long been a favorite method of statisticians. *Matchp* represents an attempt to bring the power of this method to bear on the visual correspondence problem. Despite using the simplest method of combination imaginable, *matchp* attains a rather high level of performance and so argues strongly for the applicability of this statistical tool to the correspondence problem.

Acknowledgements

Tomaso Poggio and Keith Nishihara provided important guidance. R.J. Woodham and Eric Grimson made the stereo photographs available.

Figure 2. University of British Columbia from the air

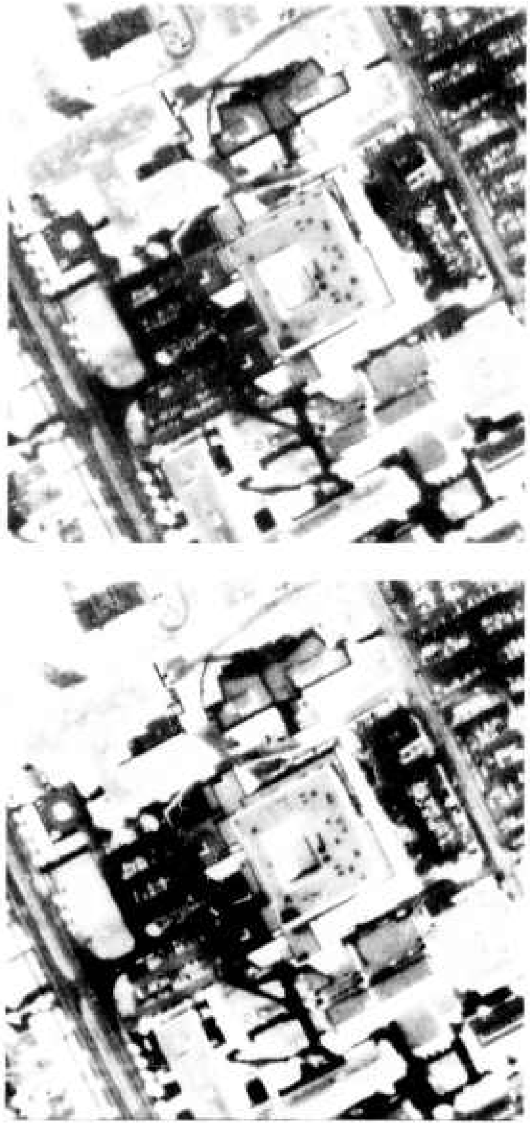
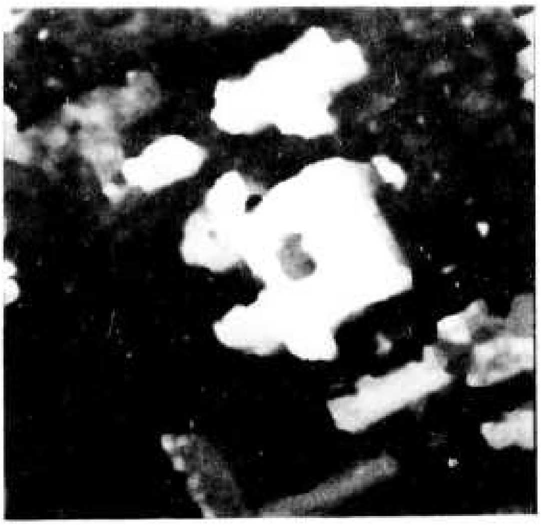


Figure 3. *Matchp* output. Intensity is proportional to depth.



References

[1] Baker, H.H. and Binford, T.O. *Depth from edge and intensity based stereo* **Proceedings of the Seventh International Joint Conference on Artificial Intelligence**, 1981, 631-636.

[2] Baker, H.H. *Depth from edge and intensity based stereo* Stanford Artificial Intelligence Laboratory Memo AIM 347, 1982.

[3] Gennery, D.B. *A stereo vision system for an autonomous vehicle* **Proceedings of the Fifth International Joint Conference on Artificial Intelligence**, 1977, 576-582.

[4] Grimson, W.E.L. **From images to surfaces** M.I.T. Press, Cambridge, Mass., 1981.

[5] Hildreth, E. C. *Implementation of a theory of edge detection* S.M. Thesis, Department of Computer Science and Electrical Engineering, M.I.T., 1980.

[6] Kass, M. *An intensity based approach to the visual correspondence problem* B.A. Thesis, Department of Independent Concentration, Princeton, 1982.

[7] Marr, D. and Hildreth, E.C. *Theory of edge detection* **Proceedings of the Royal Society of London** B 207, 1980, 187-217.

[8] Marr, D. and Poggio, T. *A theory of human stereo vision*, **Proceeding of the Royal Society of London** B 204, 1979, 301-328.

[9] Moravec, H.P. *Towards automatic visual obstacle avoidance*, **Proceedings of the Fifth International Joint Conference on Artificial Intelligence**

SMOOTHING OPTICAL FLOW FIELDS

Kwangyoen Wohn
Huchen Xie
Larry S. Davis
Azriel Rosenfeld

Center for Automation Research
University of Maryland
College Park, MD 20742

ABSTRACT

This paper describes two motion estimation algorithms. The first makes use of the scattergram of motion vectors to guide local smoothing, while the second is based on a multiresolution ("pyramid") image representation.

1. INTRODUCTION

In this paper we describe two algorithms for motion estimation. The first is an adaptation of the grey level enhancement algorithm called "superspike" to the problem of motion estimation, while the second is a multiresolution (i.e., pyramid) motion estimation algorithm. Section 2 describes the motion superspike algorithm (more details can be found in Xie et al. [1]), and Section 3 presents the multiresolution algorithm (more details can be found in Wohn et al. [2]).

2. MOTION SUPERSPIKE

This section describes an adaptation of an image enhancement algorithm called the "superspike" algorithm to motion field enhancement. In contrast to most motion estimation and enhancement algorithms which rely solely on local information in the image, the superspike algorithm utilizes global information about the motion field, derived from a histogram of the x and y components of the estimated motion. The incorporation of global information leads to both more accurate and precise estimates of motion.

Superspike was introduced in [3] as an enhancement algorithm for grey scale images; a generalization to color was presented in [4-6]. Superspike is an iterative algorithm which at each iteration replaces the grey level (or, more generally, spectral vector) of a pixel, P, by the average grey level of a subset of the pixels in some fixed size neighborhood of P. A neighbor Q is included in this averaging subset if:

1) the grey level gQ at Q is in the same histogram cluster as the grey level gP at P (this is ordinarily determined by checking that the histogram values between gP and gQ are monotonic; some smoothing of the histogram is necessary to avoid being misled by local peaks and valleys); and

2) the value of the histogram at gQ is higher than that at gP - i.e., gQ is a more probable grey level in the image than gP.

Of course, each iteration of the algorithm changes the global grey level distribution, although after only a few iterations there are ordinarily not many changes in the neighborhood subsets of pixels that are used to determine the new grey levels. Empirically, the result of applying the superspike algorithm is that the grey level histogram is reduced to a small number of spikes; this, of course, makes it trivial to segment the image into homogeneous regions. For a more detailed description of the algorithm, see [3-6].

It is possible to modify the superspike algorithm so that it can be applied to motion field enhancement. We will describe the modifications necessary for applying it to enhancing motion fields where the motion is constrained to be translation in the image plane. It is also possible, in principle, to deal with image plane rotations and zooms; however, we were not successful in obtaining useful segmentations for more general motions even when analyzing carefully controlled motion sequences.

We assume that we are given a motion field, M, which specifies the x and y components (u and v) at each pixel. Since the superspike algorithm requires a relatively dense motion field, the original motion vectors are computed using a differential technique such as described in [7-9].

The u and v components of motion are used to construct a two-dimensional histogram (or scatterplot) of M, and this histogram is smoothed over 10×10 neighborhoods using simple unweighted averaging to eliminate spurious peaks and valleys. Given a point, P, in M, we choose a subset of the points in a 3×3 neighborhood of P to compute the new u and v motion components at P. This subset is chosen using the same algorithm employed by the multispectral version of superspike. Finally, the u and v components of P are averaged with those of the pixels in the selected subset. Once these new components are computed, the two-dimensional histogram is recomputed and smoothed, and the process can be iterated.

Figures 1-2 contain an example. Figure 1 shows frames 1 and 5 of a natural motion sequence. Figure 2 shows the histograms of the x- and y-components of motion both originally and after 5 iterations of motion superspike. The two (trivially segmentable) peaks in the latter histograms correspond closely to the actual (hand-calculated) motion of the cars. Further examples can be found in [1].

3. MULTIRESOLUTION MOTION ESTIMATION

In this section we present a very brief description of a pyramid-based motion estimation algorithm, and present one example of its application to a motion sequence. More details on the algorithm can be found in [2].

Given a time varying image sequence, we construct a grey level pyramid for each frame in the sequence using median sampling. The grey level pyramids are overlapped pyramids, so that each pixel at level i has four fathers at level i+1. For the middle frame and for each level of the pyramid, we compute an initial estimate of the motion using a gradient-based algorithm. The motion estimate at level i is computed by considering the set of images at level i of the pyramid as a (reduced resolution) motion sequence, assuming that the image motion is locally a two-dimensional translation, and then computing a least squares estimate of the motion at each pixel based on the normal component of motion in a neighborhood of the pixel.

These reduced resolution motion fields are then organized into an overlapped pyramid by linking each node at level i to that father at level i+1 whose motion estimate is closest to its motion estimate. The next step, which is the most crucial, segments the pyramid into subpyramids that cover the original image. The apex of each subpyramid is, in a sense described below, the most "reliable" estimate for the motion of the pixels at the base of that subpyramid (ignoring edge effects), and its motion properties (i.e., the pattern of motion estimates in a small neighborhood of the apex) are used to adjust the flows of the other nodes in that subpyramid. The apices of the subpyramids are determined as follows. A node, f, in the pyramid dominates one of its sons, s, if

1) The magnitude of the flow at s is less than the magnitude of the flow at f;

2) The second derivative, with respect to resolution, of the flow at f is less than the corresponding second derivative of the flow at s; and

3) The predictive coding error for the area of the picture around s and f is less using the motion estimate at f than it is using the motion estimate at s.

Very briefly, the motivations supporting these criteria are: For (1), if the extent of the smoothing used to compute the spatial gradient of the time-varying image is less than the extent of the motion, then gradient-based techniques tend to underestimate the image motion. For (2), if the motion at a pixel is large, then the gradient-based motion estimates as a function of spatial smoothing increase (relatively) linearly to the correct estimate, remain stable for a while, and then change unpredictably. The second derivative of flow with respect to the resolution will be large initially, smallest during the period of stability, and then large again. For (3), since the gradient-based motion estimation algorithms assume that the grey level is an invariant to the motion, the motion field should, in principle, be a perfect estimator of intensity in subsequent frames.

This notion of dominant nodes naturally leads to a segmentation of the pyramid into subpyramids whose apices are the ends of the longest chains of dominant nodes (starting from the level above the base). Once the apices have been determined, a top-down process in each subpyramid adjusts the motion estimates at all nodes in the subpyramid. At each level (and starting at the apex) and for each node at that level we compute the divergence and curl of the motion field in a neighborhood of the node, and then adjust the motion estimates of the sons so that the neighborhoods of the sons have the same divergence and curl as that of the father. For details see [2].

Figure 3 contains an example of the multiresolution algorithm applied to the motion sequence in Figure 1. Since this sequence contains two objects moving at very different speeds, the motion estimates that are obtained using a fixed resolution are not equally reliable for both objects (the speed of the faster object is consistently underestimated, and the directions are very unreliable). Figure 3a shows the original motion estimates and Figure 3b shows the results using the algorithm.

REFERENCES

1. H. Xie, K. Wohn, L. Davis, and A. Rosenfeld, Optical flow field smoothing by local use of global information, University of Maryland Technical Report CAR-TR-3, CS-TR-1274, April 1983.

2. K. Wohn, L. Davis, and A. Rosenfeld, Multiresolution motion estimation, University of Maryland Technical Report, in preparation.

3. K. A. Narayanan and A. Rosenfeld, "Image smoothing by local use of global information," IEEE Trans. SMC, 11, 826-831, 1981.

4. L. Kitchen, M. Pietikainen, C. Wang, and A. Rosenfeld, "Multispectral image smoothing guided by global distribution of pixel values," IEEE Trans. SMC, to appear.

Viewframes: A Connectionist Model of Form Perception

Lydia M. Hrechanyk and Dana H. Ballard

Computer Science Department
University of Rochester, Rochester, NY 14627

Abstract

The aim of this paper is to show that a wide variety of perceptual phenomena have succinct explanations given the concept of *frame primitives*. A frame primitive is a local coordinate frame attached to certain more primitive perceptual data. The idea of assigning coordinate frames to objects and the role of such frames in gestalt phenomena is appreciated but has not been extensively modeled. We show that an implementation of frame primitives in a parallel, connectionist model has special virtues in understanding many aspects of perceptual dynamics.

1. Introduction

This paper has two objectives: to provide a general model for the problem of form perception and to describe the model in terms of a connectionist formalism.

Why Connectionism?

Computational models of vision have stressed the *description* of shapes, rather than the *perception* of shapes. The first problem tends to focus on the invertibility of the representation, i.e., the reconstruction of the shape's surface from the underlying description. The second problem centers around the computability of the representation, particularly in the presence of noise and occlusion.

If computability is stressed, then the choice of machine architecture becomes paramount. Much of form perception has been cast in terms of information processing models, almost exclusively based on the notion of computation as that carried out by a sequential Von Neumann machine [Ballard and Brown, 1982]. However, this model has many drawbacks as a model of human perception. Animal brains do not compute like a conventional computer. Comparatively slow (millisecond) neural computing elements with complex, parallel connections form a structure which is dramatically different from a high-speed, predominantly serial machine. Much of current research in the neurosciences is concerned with tracing out these connections and with discovering neural unit responses to complex stimuli. However, a crucial next step is to characterize neural function at a higher level than single units. Earlier connectionist models [Hebb, 1949; McCulloch and Pitts, 1943; Rosenblatt, 1958] were a step in this direction, but at the time those ideas were formed, the knowledge of the brain was much less than it is now.

Connectionism is the only current model that can stand the crucial test of timing. That is, given that entire behavioral responses can be realized in 100 milliseconds, connectionism seems to be the only way to construct plausible models in terms of neural units that can achieve these response times. Previous papers have suggested how connectionist theories of the brain can be used to produce testable, detailed models of interesting behaviors [Feldman, 1981a; Ballard, 1983; Feldman and Ballard, 1982]. These, and work by Hinton [1981a; 1981b] and Fahlman [1979] have served to shed light on connectionist architectures, but knowledge of the potential of such constructs is still in an embryonic stage. By tackling hard problems such as form perception we hope to shed light on both form perception and connectionist models.

Distributed Computation

Shape description has favored centralized representations. Examples of such work are generalized cylinders [Agin and Binford, 1976; Kanade, 1981; Shani, 1981], spherical harmonics [Schudy, 1982], 3-d generalizations of the medial axis transformation [Badler and O'Rourke, 1977], and polyhedral models [Brown, 1981]. Usually these representations are described with respect to a single, orthogonal frame. The exception is the curvilinear frame used in generalized cylinders. In constrast, we are interested in recognizing complex objects, whose representations are *decomposable* and *distributed* among many related coordinate frames. In fact, we adopt a radical view: most of our representation of shape consists *only* of coordinate frames. The approach of a *distributed structural description* has been defended in the psychological literature [Hinton, 1979; Palmer, 1977] and widely used (see, for example, Marr and Nishihara [1978], Shapiro et al. [1982], and Brady and Wielinga [1978]). More specifically, the representation is a set of shape constraints which, when combined, specify a particular form.

The representation of a shape as a set of distributed constraints has several advantages: (1) a large number of different shapes can be represented compactly, owing to the combinatorics of the distributed constraints; (2) the shape can be quickly computed by the parallel propagation of partial constraints; and (3) partial constraints can be computed independently. In addition, our connectionist model meshes well with such a representation since the architectural structure prefers distributed constraints, and the computational method is naturally insensitive to occlusion and noise.

5. M. Pietikainen and A. Rosenfeld, "Multispectral image smoothing by local use of global information," Univ. of Maryland Computer Science TR-1113, October 1981.

6. C. Wang and L. Kitchen, "Improvements in multispectral image smoothing," Univ. of Maryland Computer Science TR-1152, March 1982.

7. C. Cafforio and F. Rocca, "Tracking moving objects in TV images," Signal Processing, 1, 1979, 133-140.

8. J. Limb and J. Murphy, "Estimating the velocity of moving images in TV signals," Computer Graphics Image Processing, 4, 1975, 311-327.

9. R. J. Schalkoff and E. S. McVey, "A model and tracking algorithm for a class of video targets," IEEE Trans. PAMI, 4, 1982, 2-10.

10. B. K. P. Horn and B. G. Schunck, "Determining optical flow," Artificial Intelligence, 17, 1981, 185-204.

11. K. Wohn, L. S. Davis and P. Thrift, "Motion estimation based on multiple local constraints and nonlinear smoothing," Pattern Recognition, to appear.

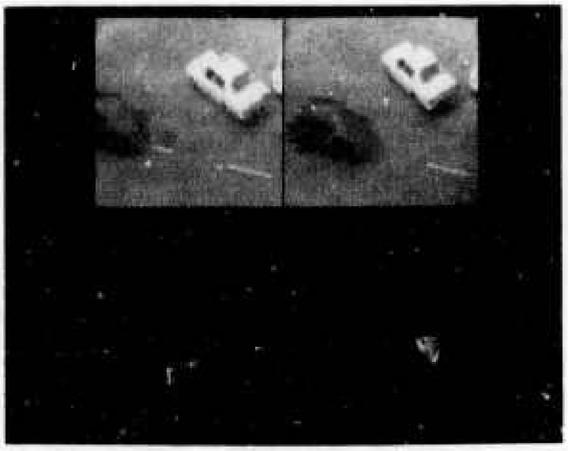


Figure 1

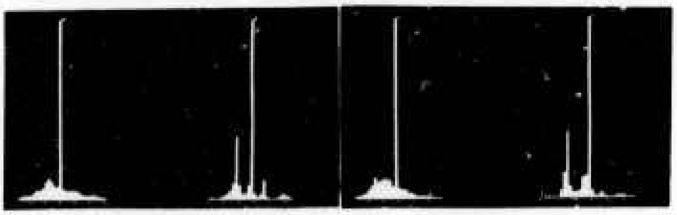


Figure 2

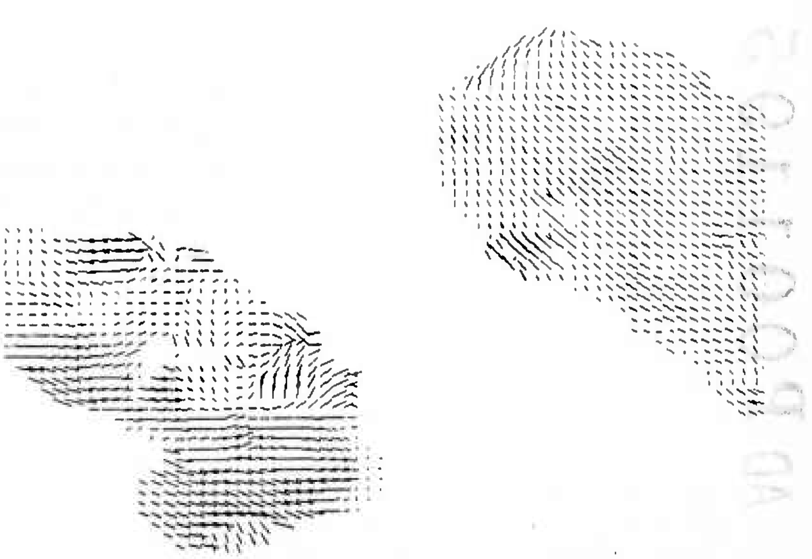


Figure 3a

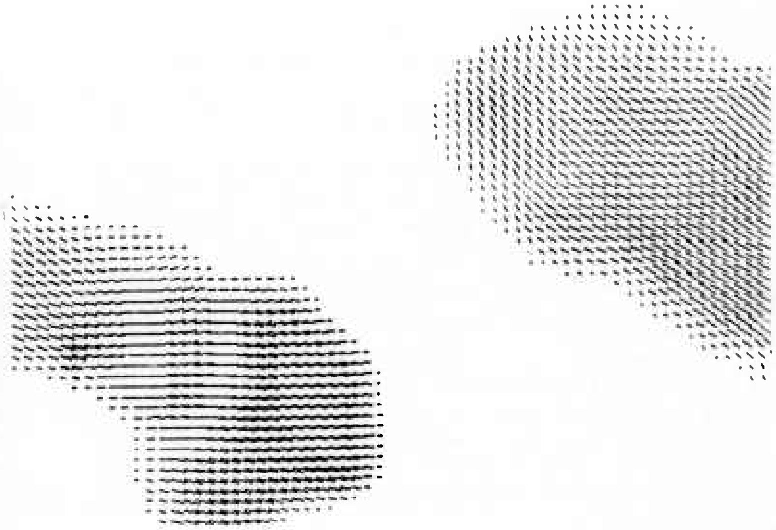


Figure 3b

Another advantage of viewframes is that they can be (at least in principle) computed directly from surface markings of lines and points and do not depend on assumptions of continuity and smoothness. Thus the representation avoids the objections raised about intrinsic images by [Witkin and Tenenbaum, 1983].

The language for expressing the shape model is that of parameter nets [Ballard, 1983; Feldman and Ballard, 1982]. The distributed, connectionist model consists of networks of the following entities:

(a) *Viewframes.* These units represent possible viewer-centered coordinate frames for perceiving a shape.

(b) *Model-frames.* A pattern of activity in these units represents a possible object-centered frame that is used to describe a form.

(c) *View parameters.* These units represent values of scale, rotation, and translation that are appropriate for specifying the relationship between the viewer-centered and object-centered frames ((a) and (b)).

(d) *Model-identifier nodes.* These units represent object tokens such as "horse," "horse's back," "bicycle," etc., in a non-geometric relational description of object structure.

(e) *View-stable features.* These units represent shape parameters that are relatively independent of small changes in the viewing frame (a). Examples would be the blocks-world joint types used by Kanade [1981].

(f) *Focus-of-attention parameters.* These units represent discrete changes of attention between model-identifier nodes. For example, a focus of attention (FOA) node might switch attention from the horse's back (= back node high confidence) to the horse's neck.

(g) *Change-of-view parameters.* These units provide the basis for changing the view transformation (c).

(h) *Shape parameters.* These units specify local shape information relative to frame parameter units. Many choices of these parameters are possible since these are the parameters encountered in holistic shape representations.

These parameter sets form a basis for describing a model of human shape perception. The next step is to describe the constraints between them. The fundamental premise is that these constraints must be *restricted to subsets of entities.* The practical reason for this is one of combinatorics; if constraints involving many different kinds of units are allowed, it becomes impossible both to represent them as connections and stay within the biologically plausible limitation of 10^4 connections per unit. However, a key point is that a set of restricted constraints, when taken together, can imply a larger constraint. The constraint relations we will need are the following:

(A) View-Transformation (viewframes, model-frames, view parameters). This relationship is the geometric transformation that relates the two different frames of reference.

(B) Relational Constraints (model-identifiers, model-frames). These kinds of constraints are those explored by Shapiro [Shapiro et al., 1982] that relate model-identifiers to ranges of model frame units.

(C) Characteristic Views (view-stable features, model-identifiers, view parameters). This relationship relates shape primitives directly to model identifiers without involving detailed geometry. For example, if we know we are looking at the side of a horse, we can expect certain shape features [Feldman, 1982].

(D) Focus-Switching (model-frames, FOA parameters, change-of-view parameters). This relationship specifies possible changes of focus. For example, any frame unit in (b) may become the viewframe by activating the appropriate change-of-view parameter.

(E) View Transformation Switching (view parameters, change-of-view parameters). This relation handles the view transform part of (D).

The ensuing sections develop the motivation for these choices of entities and relations. While this set of constraints is constantly undergoing revision, we think it provides a workable taxonomy with which to explore interesting issues in shape perception.

Outline

The discussion of these constraints starts from the most pr .nitive elements; subsequent ideas are presented in c.der of increasing complexity. For simplicity, the examples are limited to two dimensions, but this is not too serious a limitation. The 3-d versions for some of the relations have already been developed [Ballard and Sabbah, 1981; Marr and Nishihara, 1978] and the 2-d results are applicable to boundary contours, an important subcase of the general 3-d problem. We first discuss the concept of frame primitives which we term *viewframes.* Viewframes can be extracted from image data by simple rules combined with relaxation [Zucker, 1980] and Hough techniques [Ballard, 1983]. An important adjunct to the viewframe concept is that of *space-time processing.* Rather than having separate computations for spatial and temporal patterns, they are combined in a single processing network with resultant space savings. Next we describe the concept of a *view transform.* The view transform has been described in computational terms as a generalized Hough technique and has been put into connectionist terms by Hinton [1981c]. The view transform is an important kernel of any computational shape perception model as it economically relates abstractions of image data (viewframes) to model frames. Further economics arise because the view transform has an important decoupling property. Two of its four parameters, scale and rotation, can be computed independently of the other two, x-translation and y-translation [Ballard and Sabbah, 1981]. This leads to the concept of representing the view transform in a connectionist model as *split parameter spaces* (parameter subspaces).

The geometric constraints captured by the view transform are insufficient to relate any subset of image data to one of a large competing set of stored models. The rest of this paper explores solutions to this problem. The simplest is the concept of using the view transform in special modes which are more constrained than the general case. We describe *top-down mode* (looking for a given object), *bottom-up mode* (identifying a segmented object), and *tracking mode* (looking at a segmented object over time).

Since the view transformation is underconstrained, a natural solution is to find additional constraints that are not geometry-based (e.g., color), and that can be derived from the image independently. Many possibilities are discussed in [Ballard, 1983; Feldman, 1982]; we will not pursue these possibilities here. The most ambitious solution to the underconstrained nature of the view transform is to find some *hierarchical indexing scheme* for the model shapes so that they do not all have to be considered at once. Such a scheme requires a sequential scanning mechanism to move back and forth between levels of abstraction. We describe such a mechanism in terms of the formalism and relate it to Iarbus's scanning results with human subjects [Iarbus, 1967].

2. Viewframes

Representations for geometrical objects are usually greatly simplified if an appropriate coordinate frame is chosen. The case is even stronger for articulated objects [Marr and Nishihara, 1978]. The fact that geometric representations simplify if appropriate coordinate frames are chosen agrees well with the many human perceptual results that suggest frame-dependencies (e.g., [Hinton, 1979]).

Computing good coordinate frames for complex objects is in general difficult, although some progress is being made. Brady [1983] shows that the boundary features of objects suggest logical possibilities. Also, he argues that rather than a single frame, one should think of a hierarchy of possibilities, where different levels in the hierarchy depend on different levels of surface detail and locality. (Similar points were made in [Marr and Nishihara, 1978], but the emphasis was less on computability.)

While the notions of coordinate frame are intuitive for segmented objects, it may be less obvious that they are an essential component of the descriptions behind a variety of gestalt grouping phenomena (Figure 1). Our model for all these phenomena depends on frame primitives, which we have termed viewframes. Viewframes may be strongly suggested by shape contours (e.g., [Brady, 1983]) or they may be only weakly suggested by primal tokens. In these latter cases, viewframes are characterized as being the essence of the description, rather than an indexing mechanism for surrounding complex geometry.

The crucial issues in describing viewframes are how they are represented and how they are computed. Besides issues of abstract computability, it is important that viewframes be computable in terms of connections. Consider the 2-d case: a complete set of parameters is specified by the location of an origin x,y, a rotation θ, and a scale s, and these parameters describe the relation of a local viewframe to a global image frame. This 4-d space of parameters covers all possible local frames. To represent this space, discretize it using some Δx, Δy, $\Delta \theta$, and Δs, and assign each resultant discrete cell a value unit. (Ways of economizing on the number of units will be described in Section 3.)

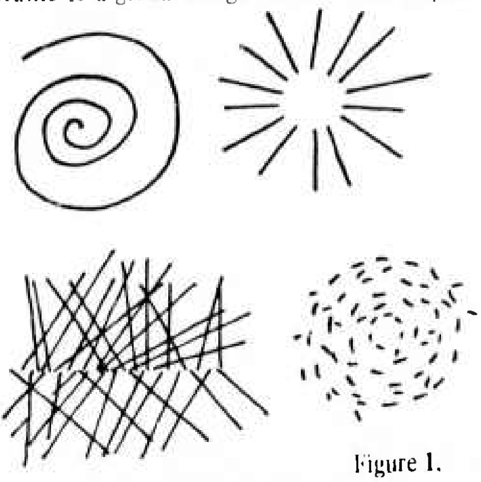


Figure 1.

Frame space can represent all the frames that could be present (at the resolution level chosen), but only a fraction of those will usually be present in a given image. Furthermore, of these, many can be ruled out as being inconsistent on the basis of local and global frame grouping rules. Thus

representable frames > possible image frames
> consistent image frames

In the ensuing paragraphs we will describe the process of computing consistent image frames in more detail. In the process we attempt to synthesize a unifying explanation out of our own previous work [Ballard, 1981; Ballard and Sabbah, 1981] as well as that of others [Stevens, 1981; Zucker, 1980]. The reason for attempting a synthesis is that the frame description is necessary as a primitive for all our subsequent work, and that important precursors have appeared that do not explicitly acknowledge frames [Zucker, 1980] or that (from our point of view) miscast the frame assignment problem as a correspondence problem.

Rules for Frame Suggestion

We assume that the visual environment has been tokenized in some way, e.g., collections of points and edges that may or may not be moving. Out of this primitive structure, the next useful level of abstraction concerns the suggestion of frames. The initial suggestion of frames corresponds to initial levels of activation of units in frame space. These levels may be raised or lowered depending on the surrounding context of nearby frames. The rules for frame suggestion are summarized in Figure 2. These rules are based on a 2-d model.

A *curve* has a natural local frame which is its tangent. The tangent specifies a frame origin locus and orientation locus, leaving scale undetermined. With this case and the others to follow, two orientations of the frame are possible, one at θ and the other at $\theta+\pi$.

Line segments are similar to curves except that there is a natural choice for scale which is the length of the line.

Points suggest a frame origin but leave scale and rotation undetermined.

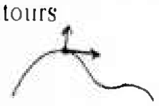
Moving points have a natural frame parallel to the direction of motion. The velocity suggests a value for scale.

Two points have a natural frame whose orientation and scale are determined by the line joining the points and whose origin is located at the leftmost (with respect to the frame) point.

Where the tokens are not the aforementioned primitives, but rather complex shapes, these rules may still apply between the local frames of the more complicated shape tokens. Complex shapes seem to have their own rules for their local frames [Brady, 1983].

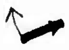
Rules for Frame Assignments

a) Contours



x,y,θ defined by curve; s undetermined

b) Line Segments

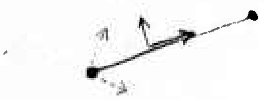


two possible choices of x,y,θ; s determined

c) Points

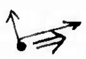
x,y determined; s,θ free

d) Long Line Segments



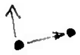
long line segments can be viewed as having points at their ends; rule for point frames applies

e) Moving Points



moving points have natural frame parallel to direction of motion; curvilinear case is analogous to moving along boundary contour; x,y,θ determined; s free

f) Correspondence



two points have natural constraint of frame that aligns with line joining points

Figure 2.

Frame Relaxation

Frame units have an associated activation level which ranges between zero and one. An activation level signifies whether or not the frame unit is part of the current gestalt. Thus a frame unit which is initially activated by lower level input may have its activation reduced if it is inconsistent with neighboring frame units in its surround. Methods for increasing or decreasing activation have been previously developed. Zucker [1982] has exactly the right kind of algorithm for refining frames based on purely local evidence. In that multilevel relaxation scheme, pairs of points suggest line segments (rule f) and nearly colinear line segments can increase each other's activation level. (To translate from relaxation labeling to connectionist relaxation, make a unit for every label and let the probability of a label be its activation level [Ballard, 1983].) Similar methods based on correspondence have been used [Ullman, 1979; Barnard and Thompson, 1979], but correspondence leads to problems if taken too literally, since non-correspondences owing to noise and occlusion have damaging effects.

The best way for dealing with local frame coherence is to look at the mode of the distribution of local frame parameters [Stevens, 1981]. This allows frame coherence to emerge from high levels of ambiguity. Cognoscenti will recognize this method as a version of the Hough transform [Ballard, 1983].

Besides local frame coherence, there is also the global frame coherence found, for example, in glass patterns. If identical spotted overlays are rotated a small amount with respect to each other, a global concentric pattern is seen. This global pattern can be explained by postulating a parameter space that explicitly represents parametric variations in the pattern. Each local frame raises the activation of units in the parameter space that are compatible with itself. In this case the global units represent rotation center coordinates and the local frame raises the activation level of (metaphorically: votes for) parameter units on a linear locus perpendicular to the x-axis of the frame (see Figure 3). Note that the Hough transform model for computing the rotation center provides a mechanism for selecting the *mode* of the activated units. The many spurious activations that arise from suggested frames that are not part of the global pattern are spread among very disparate units, and thus can be discounted via local inhibition. Other methods that describe this construction (e.g., [Hildreth, 1983]) do not acknowledge the above problem in applying it.

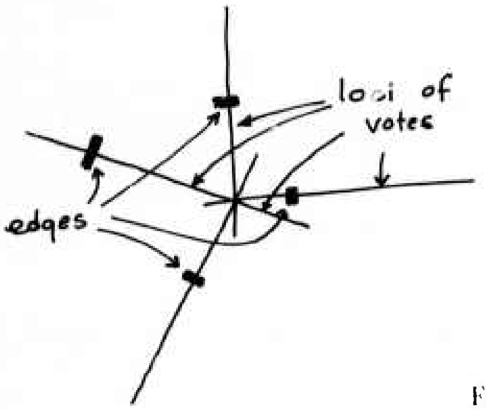


Figure 3.

3. Space-Time Processing

A viewframe is an abstraction that can arise from more primitive stimuli which may be either spatial (e.g., rule f) or temporal (rule e). Thus the problem of recognizing patterns in collections of frames can be cast as one of abstract geometry independently of whether the patterns arise from spatial or temporal data. An example should make this clear: consider radial lines emanating from a common point (Figure 4). These may arise from the common point of projected parallel lines, a vanishing point, or from the common focus of expansion of optic flow due to a a translating observer. These spatial and temporal phenomena are closely related; the loci of points translating with respect to a rectilinearly moving observer are also parallel lines.

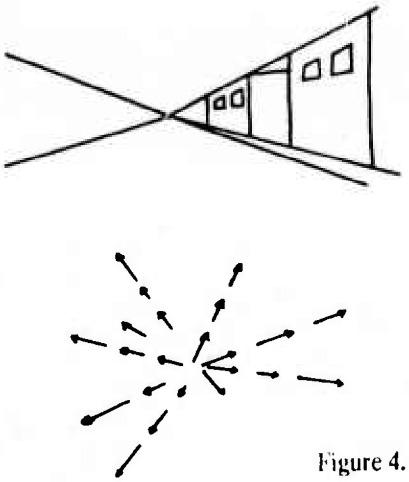


Figure 4.

The global transformation to detect radial lines can be carried out in two steps. The first detects the colinearity of oriented frames and the second detects the pattern, in line parameter space, which is due to the radial field. In the notation for parameter transforms [Ballard, 1983], we can describe the line transform as

$$\langle (x,y,\Delta x,\Delta y),\ (r,\theta),\ (\theta = \tan^{-1}(\Delta y/\Delta x);\ r = x\cos\theta + y\sin\theta)\rangle.$$

The parameters Δx and Δy describe the direction of the frame vectors at a location x,y. The notation <( ),( ),( )> means that the units described by the parameters in the first set of parentheses will raise the activation levels of a subset of those in the second set of parentheses. The relationships in the third set of parentheses describes the subset.

Radial lines map into circles in (ρ,θ) parameter space and these can be detected by

$$\langle (r,\theta),\ (a,b),\ (r/2 = a\cos\theta + b\sin\theta)\rangle$$

Note that at this point, whether $(x,y,\Delta x,\Delta y)$ arises from intensity gradients or flow vectors has been left indeterminant. Of course in order to use the answer effectively, one must know whether the computations are relevant to space or time.

Some of the results that can be computed from a frame processor are valid for both space and time, and others are only valid for either one or the other dimension. For example, the distance of closest approach, Q (given by Eq. 7.1.3 in [Ballard and Brown, 1982]),

$$Q^2 = (\mathbf{x}\cdot\mathbf{x}) - (\mathbf{x}\cdot\mathbf{O})^2/(\mathbf{O}\cdot\mathbf{O})$$

where

$$\mathbf{O} = (a,b,1) \text{ and } \mathbf{x} = ((f-z)x/f,\ (f-z)y/f,\ z)$$

is valid for both space and time, but "time-to-adjacency," given by the HT

$$\langle (a,b,x,y,\Delta x,\Delta y),\ (t),\ (t = d/||v||)\rangle$$

where

$$||v|| = \sqrt{(\Delta x^2 + \Delta y^2)}$$

and

$$d = \sqrt{((x-a)^2 + (y-b)^2)}$$

is only valid for the temporal interpretation.

As another example, consider the detection of spiral patterns. Since these can be perceived on the order of 100 ms, like glass patterns, our hypothesis is that the perceptual mechanism must be manifested as connections (see also [Feldman and Ballard, 1982]). However, spiral patterns that arise as strictly spatial patterns in nature, while possible, are rather infrequent. In contrast, spiral patterns derived from temporal loci are frequent experiences. For example, as discussed earlier, the optic flow due to an observer translating is radial. This flow, summed with the concentric flow produced by a rotation about the direction of travel, leads to spiral temporal patterns. Note that the flow is present over the full visual field, even for a short temporal duration. Thus the frame processor architecture implies that the ability to recognize infrequent spatial spirals is a direct consequence of the ubiquitous nature of temporal spirals that have the same underlying geometry.

Weak corroborative evidence for such transfer comes from the Fraser illusion (Figure 5). In this illusion, spirals are seen even though the global patterns are concentric circles. Presumably this is because the combined, local evidence predicts spirals. This is precisely the effect one would expect from temporal spirals as the predominant experience would occur in spatio-temporally local segments.

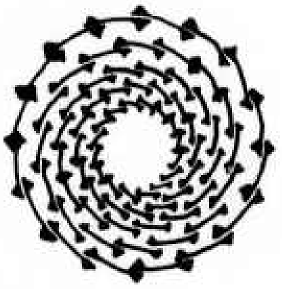


Figure 5.

The principal virtue of having a frame processor is that its circuitry, together with that necessary to distinguish between space and time, is much less than that required to implement two independent processors, one for space and one for time.

One of the problems that might occur with a single frame processor is that separate spatial and temporal events could be confused. This may indeed happen. For example, in the Pulfrich pendulum illusion, a point of light oscillating in the frontal plane is viewed with two lenses, one of which has been darkened. In this illusion, the darker input is interpreted as a temporal delay, and the point is seen to move in depth. For most cases, however, the modes of use of the frame processors can be separated into different spatio-temporal regimes which do not overlap.

4. Computing the Viewing Transformation

An object's viewframes may be related to its internal frame representation by a *viewing transformation*. Knowing any two of: {the internal representation, the viewing transformation, and the viewframes} allows the third to be computed. Usually the viewer-centered data are known but both the corresponding internal shape and viewing transformation must be computed. This problem is generally underdetermined [Palmer, 1981]. Furthermore, the image data is usually cluttered with many features that belong to different objects, and these tend to confound the perception of a particular shape. Previous work [Ballard, 1981; Ballard and Sabbah, 1981; Sloan and Ballard, 1980] made the simplifying assumption that the internal representation contains only a single object. In this case the viewing transformation could be computed and parts of the object in the image identified despite other image clutter. The task of determining if a known object is in an image is posed as: is there a transformation of a subset of image features such that the transformed subset can be explained as the object? If the answer to this question is no, then the object is not present. If yes, then the transformation provides all the necessary information about the object. In a connectionist network, the affirmative answer is represented by the convergence of a view transform network to a simple active unit.

The viewing transformation is completely specified in the 2-d case by four parameters: two for translation; one for orientation; and one for scale. Ballard and Sabbah [1981] have shown that it is possible to decouple the interdependence of two subgroups of the parameters for scale and orientation from translation. In other words, the orientation and scale of the object can be detected without knowing its translation. In fact, there is a natural precedence of parameters:

$$\text{scale} > \text{orientation} > \text{translation}$$

This precedence stems from different factors. The reason scale is simple to detect is that it is available from *intrinsic image* data [Barrow and Tenenbaum, 1978]. An intrinsic image is an image of some important parameter that is retinotopic; that is, in registration with the intensity data on the viewer's retina. For scale computations the most important image is the *depth map* [Marr and Nishihara, 1978]. A depth map represents distances with respect to the viewer. Thus if the internal representations have an associated absolute metric the scale between an object-centered feature and a viewer-centered feature can be immediately determined. Orientation is easier to detect than translation as it is functionally independent from it, whereas the reverse is not true. In other words, viewer-object orientation correspondences can be computed without considering translation, but to do the same for translation correspondences, one must know the appropriate values of orientation and scale.

Representing the View Transform with Connections

The connections for the view parameters may be viewed as a form of Hough transform using *constraint tables* [Ballard, 1981]. Matches between image frames and object frames constrain the values for the viewing transform parameters. Each image frame maps into only a set of allowable parameter values. When all the frame matches are taken into account, the mapping is many-to-one onto plausible parameter values. (Since the cost of this method is exponential in the number of parameters considered together, the decoupling of parameters into groups of scale, orientation, and translation mentioned above is very significant [Ballard, 1983], and we will pursue this in a moment.)

In this paper we will restrict our examples to two dimensions, although the constraints for the 3-d case are only slightly harder [Ballard and Sabbah, 1981]. Consider a 2-d primitive specified by a single x-axis vector **x** defined in the viewer frame. Suppose it corresponds to a vector y in the object frame. The transformation between x and y is specified by view parameter p where $\mathbf{p} = (\theta, \Delta x, s)$. These parameters correspond to orientation, translation, and scale, respectively. In the parameter network a particular unit p will receive *conjunctive connections* from appropriate pairs of units x and y where

$$\Delta x = x - y$$

$$s = ||x|| \; / \; ||y||$$

$$\theta = \text{angle}(x) - \text{angle}(y)$$

where "angle" is the angle between the vector and the x-axis in the appropriate frame

The foregoing description specifies approximately one third of the connections implied by the view transform relation (A): view parameter units receive connections from model units and frame units. However, as Hinton [1981c] has pointed out, the role of the entities in relation (A) is symmetric, and each may receive connections from the other two. For example, a unit x will receive conjunctive connections from units y and p where

$$x = y + \Delta x$$

$$||x|| = s \; ||y||$$

$$\text{angle}(x) = \theta + \text{angle}(y)$$

The conjunctive connections are used to specify that several parameters logically need to be present simultaneously in order to effect the behavior of a unit. Where the connections between networks are symmetric, we will use Hinton's notation of a small triangle at the junction of connecting lines (see Figure 6).

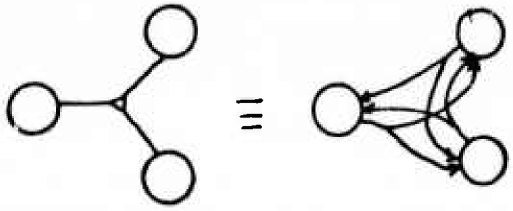


Figure 6.

Confidence Updating Strategy

Given the input connections to a unit, which may be conjunctive, one still has to specify how to update the confidence of the unit. There are many reasons for not using a strict summation [Feldman and Ballard, 1982]. We use a "normalized maximum of sums over threshold" formula. Let the inputs to a unit be organized into sets of conjunctive connections $\{S_i\}$ and let each such set have a threshold θ_i. Let C_i be the sum of the confidences of the units in S_i. Then the confidence of a unit j is updated by:

$$C_j = \max\{S_i - \theta_i\} / (\max\{C_j\}).$$

The behavior of the numerator is easy to accommodate in the behavior of a unit, but the denominator requires a separate network, such as those described in [Feldman and Ballard, 1982].

Split Parameter Spaces

The connectionist implementation of the view transform computations up to this point has utilized the constraints developed in [Ballard and Sabbah, 1981] and is a variant of Hinton's letter recogition model [Hinton, 1981c]. However, this approach requires too many units. Consider the 2-d case. If we allow 100 values for each of scale, orientation, and horizontal and vertical translation, each network in the view transform requires 100^4 units. More problematic is the approximately 100^4 conjunctive connections per unit, which is totally unrealistic. Hinton has suggested reducing the units by using units with overlapping parameter values [Hinton, 1981b]. This concept reduces the number of units by a factor of 1/D)$^{k-1}$ where D is the diameter of the unit and k the dimension of the parameter vector. While this is a dramatic reduction and biologically plausible, it may still not reduce the number of units enough, and it places an added burden on the number of conjunctive connections required. A complementary way of reducing the number of units required is to use *split spaces*. Split spaces is the concept of representing a high-dimensional set of units with subsets of units of lower dimensionality. For example, the 4-d model frame net can be represented as two networks, one with location units (x,y) and one with length and orientation units (l,θ). Split spaces introduce the possibility of erroneously associating parameter subsets but the probability of a false association can be made extremely small with the assumption that the space is in some sense "sparse" (i.e., the number of units active at any one time is not too large).

Figure 7 shows the split space representation of the 2-d viewer transformation computation. The key point is that the computation be made sequential to the degree required by dependencies. Thus, translation parameters are not computed until the scale and rotation parameters have been found. The process is dynamic in that these latter parameters can in turn indirectly effect the previous ones. The network contains the following three groups of constraints:

(1) *The connections for scale and orientation.* As before, one can use conjunctive connections to determine the relationship between model-frame length and orientation, image-frame length and orientation, and the corresponding view parameters. The difference is that these units represent only length and orientation parameters and do not involve translation.

(2) *The connections for rotated model parameters.* The key to this implementation of split spaces is the use of rotated model-frame units x' related to x by

$$x' = s \; \text{ROT}(\theta)(x)$$

where Rot(θ) is the appropriate rotation matrix.

(3) *The connections for translation parameters.* Since *rotated* model units can differ from image units by at most a translation, the third set of connections between units is determined by the equation

$$x' = x + \Delta x.$$

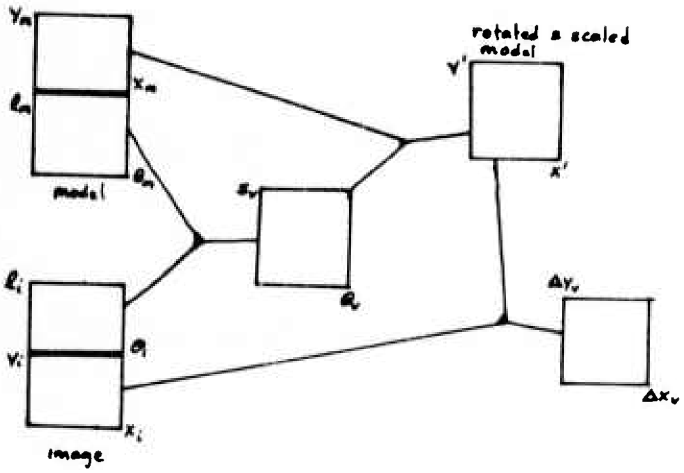


Figure 7.

Modes

In general the network of Figure 7 is not adequate to match any subset of image frames with a subset of model frames, since this problem is underdetermined. In connectionist terms, this means that if sets of model frames for many different prototypes are active, together with many different image frames, the view transform network will not converge. There are, however, many restricted cases where convergence is possible. These restricted cases would arise from connecting the view transformation kernel to auxiliary networks such as those described earlier. We can partition these cases into three important categories: top-down mode, bottom-up mode, and tracking mode. In top-down mode, a single object is being sought, and thus the model frame space contains only active units for that prototype. In this case the view transform network will converge under high noise levels (= many active non-object units) in the image frame space. In experiments [Ballard and Sabbah, 1981], up to four to five times the number of frame units could be activated before the view transform network would converge to a false mapping. In bottom-up mode, a set of frames corresponding to a single unit has been segmented, and the problem is to map that set onto one of a collection of simultaneously active prototypes. Experiments in this mode have not been done, but the symmetry of this mode with top-down mode suggests that similar results would apply. This does not mean that five prototypes could be tested for at once; rather, the union of the number of frames in all of the prototypes should not exceed four to five times the number of frames in any one prototype. Obviously a huge number of prototypes could be tested for simultaneously. Where N is the number of frames, the number of prototypes is $\binom{4N}{N}$. The third mode of the view transform network is that of tracking. In this mode segmented image frames are "transfered" to the model frame space at an acquisition time. This is done by priming the transformation network with the identity transform. Thereafter the view transform records the transformation that the image frame data undergo. Tracking mode is somewhat different than the other two modes in that the model prototype does not have to be known a priori: instead, at some acquisition time, the image frames can be used as model frames. If this prototype is to be remembered, then some mechanism must handle this. One good possibility is that of recruitment [Feldman, 1981b].

Tracking mode may also be used to recognize spatial regularities. Suppose we are given the pattern of Figure 8. Further suppose that by some fixation technique one element of this pattern can be "loaded" into the model frame space. The connection patterns will naturally compute all the view transforms between the model frame units and the other instances of the image frame units. In the case of the example, Figure 8, a linear pattern of active units will be seen in view transform space. This linear pattern explicitly captures the main part of the notion of regularity seen in the original pattern.

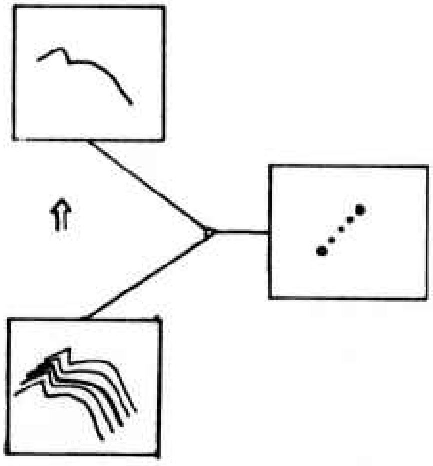


Figure 8.

5. Focus of Attention

In this section we argue two points: (1) that shape representations are hierarchical [Marr, 1982]; and (2) that levels of a hierarchy are necessarily examined sequentially.

The Need for Hierarchical Descriptions

Hierarchical shape descriptions were eloquently defended by Marr and Nishihara [1978], and our notion of such is essentially captured in their work. We do not insist on generalized cone primitives as they did, but require that whatever representations are used have a geometrical basis and defining coordinate frame. Thus, the crucial part of the representation is the *hierarchical organization of different coordinate frames.* The part descriptions with respect to those frames could take many forms, e.g., polyhedra or splines.

There are several reasons why frame hierarchies are important. First, complex objects are more simply represented by pieces described with respect to different coordinate frames than with a complicated description that uses a single frame. Second, the hierarchical organization of these frames allows for ease of indexing and the explicit representation of intermediate hypotheses. In a computational scheme for accessing the details of the parts through their more general features, hierarchical organization allows for strategies that take steps proportional to the logarithm of the number of parts. A final reason for hierarchies is that their varying "grains" form a logical description of the different viewing conditions encountered by the perceiver. When an object is distant, only gross features will be apparent owing to limitations of the imaging optics (as well as others in post processing). A proximal object may reveal details but may be so close that it cannot be imaged in a single view. In these cases the identification problem is simplified if the shape representation itself is also segmented in terms of the resolution of its parts.

Shape hierarchies can be defined relative to different complexity measures. These are closely tied to the type of *parameterization* allowed, including the aforementioned "grains" (shape distortion), connectivity relations, and the articulation freedom allowed in the parts of an object. However, there are strong dependencies between these factors when one considers how recognition proceeds. For example, in perceiving a horse, it is necessary to determine exactly where its neck is relative to its body before considering its detailed shape. In the following discussion we will be concerned mainly with the representation of articulation in objects.

The Frame Hierarchy in Connectionist Nets

The above points speak of the necessity of hierarchies but are neutral with respect to their implementation in hardware. To describe the implementation of shape hierarchies in nets, we turn first to the structure of the model identifier net.

The model identifier net receives connections from the model-frame net and also connects to it. The purpose of such connections is to implement the relation (B) described earlier. The units in the model frame are in a canonical form; that is, they are independent of the view (that variation is captured by view parameter units and the view transform). The canonical form greatly simplifies the implementation of relation (A) since, for example, a horse's back frame will always correspond to the same model unit.

In general, the connections between units in these two representations will not be one-to-one. The reason is that the model-identifier net has a relational character; a horse's neck unit in that frame corresponds to several possible units in the model frame net, owing to the fact that the neck can move relative to some fixed part in the body frame (which we choose to be the back). Figure 9 illustrates this. A second point is that units in the model frame net also receive connections from the image and view parameter networks. The intersection of units receiving excitation will help specify the appropriate corresponding model-identifier units.

We use conjunctive connections from the frame nodes in the model-identifier nets to appropriate model frame units. Thus, a model frame unit may become active only if it is receiving image input, and is in the correct "context."

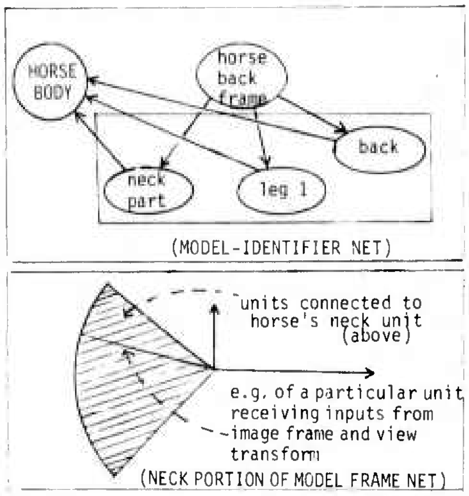

HORSE BODY
horse back frame
back
neck part
leg 1
(MODEL-IDENTIFIER NET)
units connected to horse's neck unit (above)
e.g. of a particular unit receiving inputs from image frame and view transform
(NECK PORTION OF MODEL FRAME NET)

Figure 9: A partial example of the model identifier net for a horse, and the model for one of its subparts.

Frame Switching

The most important point of the previous constraint in terms of computational complexity is that *although all identifier units are connected to model frame units, only a small portion are active at any one time.* To see the importance of this decision, consider an alternative: a separate model frame net for each identifier unit. This would allow all possible shapes to be processed in parallel but would require an unrealistic amount of units. Our principal hypothesis is that there is limited hardware to compute the view transform (here we allow only one piece of hardware) and the meaning of the active units therein is determined by the active model-identifier units connected to it.

Given that only a small set of model identifier units can be active at any one time, one needs a mechanism for switching between sets of units at different hierarchical levels. An example of such a mechanism is diagrammed in Figure 10. The first problem is to select a different frame to examine. This is accomplished by putting the model frame net into "Winner-Take-All"-mode. (For a discussion of WTA nets the reader is referred to Feldman and Ballard's work [Feldman and Ballard, 1982].) Selecting a single unit from the model frame net has three effects:

(1) it enables a frame switch unit;

(2) it deactivates all but a subset of the currently active model identifier units;

(3) it allows the appropriate view transform change to take place.

Referring to Figure 10, the **frame switch** unit connects to *all* particular frame switch units but since this is a conjunctive connection with model-identifier units only an appropriate subset of particular frame switch units will be excited. An activated frame switch unit in turn activates its corresponding frame unit, and deactivates the ancestor frame. The newly activated frame unit will then excite the model identifier unit members of its frame. We have also designed a transform switch mechanism which proceeds in lock-step with this, and allows image data coupled with current hypotheses to compute the next transform. However, due to lack of space, we omit its presentation here.

A particular advantage of this design is that evidence for a more abstract unit can accumulate even though a less abstract frame is active. Even when *horse's body* is not in the current frame, its activation can be increased indirectly through ancestor connections from an active *horse's ear* frame.

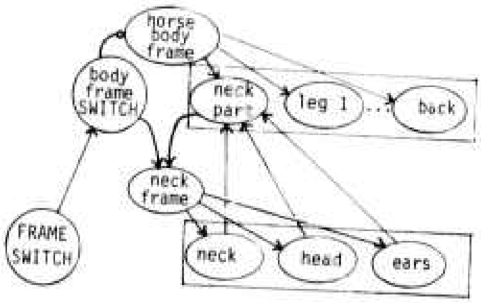


Figure 10: An example of the frame swiching mechanism, in which attention switches from horse's body to neck. Connections to model units are not shown The link with a circle at its tip represents an inhibitory connection.

Some Motivation for Frame Switching

The foregoing discussion developed the motivation for frame switching from representational grounds. In a connectionist network, the representations are necessarily distributed into pieces because of bounds on the number of connections per unit [Feldman and Ballard, 1982]. There are at least two additional arguments, however, for frame switching. One is that the mechanism can resolve ambiguities which arise from split-space representations (Section 4). The other argument is derived from psychological tests.

A problem with any split-space representation is that one cannot keep track of the correspondences between units in each subspace. This problem is shown by Figure 11, where the model frames are different from the image frames, but this difference does not show up in any of the subspace computations. One solution is to have a hierarchical representation of the shape. In the example, the primitives would be represented once as separate units and again as units which are part of a global frame. If the object is rigid, one can switch the focus to a single unit while changing the viewing transform in a predictable way. The fact that this cannot be done with the example in Figure 8 is the mechanism for detecting the mismatch.

A second indication of the importance of frame switching is due to a clever construction by Pavel (Figure 11). In this figure, three tokens are moved along linear loci. If the tokens are rotationally symmetrical (series a), the overall perception is that of a moving triangle, but if some pronounced asymmetry is given the tokens (series b), the perception immediately switches to that of three independent translations. The explanation for this perception in terms of our connectionist model uses a collection of the mechanisms already suggested. First, the tokens are analyzed by putting the view transform in tracking mode. The rotationally symmetric tokens can suggest frames (rule f) that excite a single set of view units. Even if frame switching is used, the view transform is still supported, owing to the rotational degree of freedom in a single token. In the second series, however, the situation is very different. If the tokens have a pronounced asymmetry, the corresponding frame units will predominate, and their loci are incompatible with a single set of view transform units.

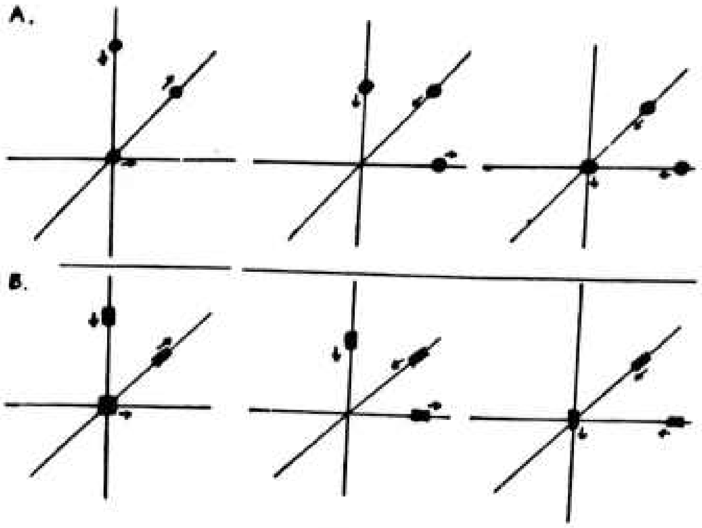


Figure 11.

One feature of our frame switching model is that it is necessarily discrete. Focus of attention switches between individual frame units that have discrete separations. Some support for this model can be derived from the classic experiments in eye movement tracking [Yarbus, 1967]. Subjects examining pictures typically used saccadic eye movements that varied as a function of task required, but more importantly:

- subjects appear to have a *fixed hierarchy of interest protocols*, e.g., humans > animate > inanimate, etc.
- subjects appear to have *fixed scanning protocols*, i.e., subjects examining the same picture on different days would exhibit essentially similar eye movement patterns.

6. Summary and Conclusions

Figure 12 summarizes the various constraints employed by our connectionist model. In this figure we have abandoned the triangle notation and only indicate the relations between nets without describing the detailed nature of the connections. This is remedied by earlier descriptions and figures.

The validity of the new transform approach has been tested in non-connectionist algorithms [Ballard, 1981; Ballard and Sabbah, 1981; Sloan and Ballard, 1980]. Currently the connectionist version of relation (A) and a frame switching mechanism similar to the one described have been successfully implemented; other networks are being programmed.

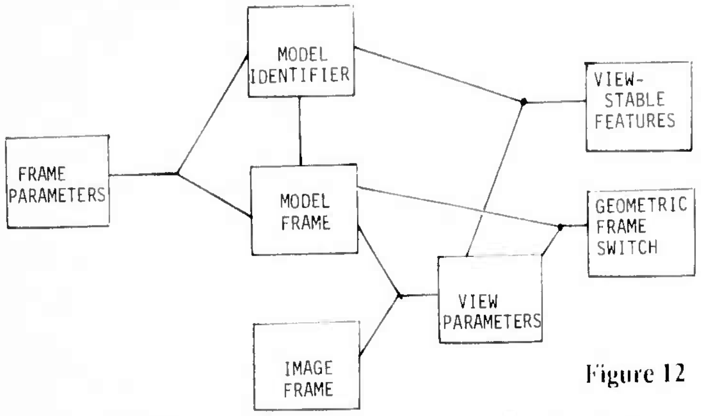
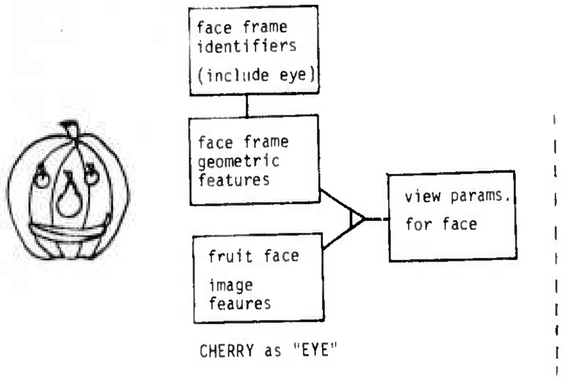
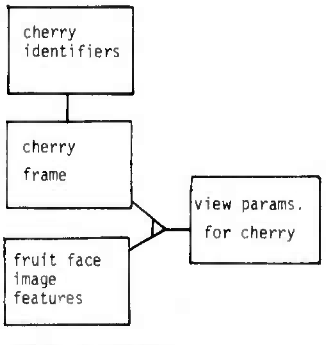


Figure 12

A difficult problem for any form perception model is to explain the perception of "fruitface" [Palmer, 1975], as shown by Figure 13. (Many similar examples can be constructed.) Detailed experimentation with our model will be necessary to determine whether it can exhibit appropriate oscillatory behavior between, e.g., seeing a cherry as a cherry and seeing a cherry as an eye. However, the representation allows for such behavior. Figure 13 shows the two states of the nets corresponding to the two different perceptions. Our use of frames is similar to Hayes' [1978], and satisfies Hinton's notion of a system that uses the same image features but assigns them different "roles" [Hinton, 1981c]. In seeing the cherry as an eye, the face frame is active and the geometric frame features for the cherry are part of active connections to the eye unit via the view transform. In seeing the cherry as a cherry, both the cherry frame and the cherry unit in the model identifier net are active.


face frame identifiers (include eye)
face frame geometric features
view params. for face
fruit face image feaures

CHERRY as "EYE"



CHERRY as "CHERRY"

Figure 13: Plausible alternatives for "fruit-face."

Many refinements of the basic approach are currently being studied. Feldman [1982] is developing a computational basis for representing objects in space in terms of four coordinate systems. This work meshes with our own in that more immediate (foveal) and more abstract (environment) frames are described as well as frames similar to our image and model frames. An incorporation of a foveal/eye movement mechanism would be of immediate advantage to the current system. By adjusting viewing parameters (e.g., centering the view on the current frame) one could minimize the units that have to be represented in the view transform.

One issue that we have sidestepped is that of the most abstract control. What triggers a frame switch? Many possibilities exist, e.g., breadth-first scanning in model-identifier space, the same in model-frame space, a "pre-wired" pattern of checking, as well as others, but the problem is still open. However, as Posner [1978] suggests, our job may be to give the homunculus less and less to do; hence our confidence in the present system.

Acknowledgements

The research described in this paper was supported in part by NIH Grant No. 2R01 HL21253, in part by NSF Grant No. MCS-8203920, and in part by DARPA Grant No. N00014-82-K-0193.

References

Agin, G.J. and T.O. Binford, "Representation and description of curved objects," *IEEE Trans. Computers C-25*, 440, April 1976.

Badler, N.I. and J. O'Rourke, "A representation and display system for the human body and other 3-d curved objects," TR, Computer Science Dept., U. Pennsylvania, February 1977.

Ballard, D.H., "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition 13*, 2, 111-122, 1981.

Ballard, D.H., "Parameter networks: Towards a theory of low-level vision," *Proc.*, 7th IJCAI, Vancouver, B.C., Canada, August 1981; to appear, *Artificial Intelligence*, 1983.

Ballard, D.H. and C.M. Brown, *Computer Vision*. Prentice Hall, 1982.

Ballard, D.H. and D. Sabbah, "On shapes," *Proc.*, 7th IJCAI, Vancouver, B.C., Canada, August 1981.

Barnard, S.T. and W.B. Thompson, "Disparity analysis of images," TR 79-1, Computer Science Dept., U. Minnesota, January 1979.

Barrow, H.G. and J.M. Tenenbaum, "Recovering intrinsic scene characteristics from images," Technical Note 157, AI Center, SRI Int'l., April 1978.

Brady, M., "Criteria for representations of shape," in J. Beck and A. Rosenfeld (eds). *Human and Machine Vision*. Academic Press, 1983.

Brady, M. and B.J. Wielinga, "Reading the writing on the wall," in A. Hanson and E. Riseman (eds.), *Computer Vision Systems*. New York: Academic Press, 1978.

Brown, C.M., "Some mathematical and representational aspects of solid modeling," *IEEE Trans. PAMI-3*, 4, 444-453, July 1981.

Fahlman, S.E., *NETL, A System for Representing and Using Real Knowledge*. Boston, MA: MIT Press, 1979.

Feldman, J.A., "A connectionist model of visual memory," in G.E. Hinton and J.A. Anderson (eds). *Parallel Models of Associative Memory*. Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers, 1981a.

Feldman, J.A., "Four frames suffice," TR 99, Computer Science Dept., U. Rochester, September 1982.

Feldman, J.A., "Memory and change in connection networks," TR 96, Computer Science Dept., U. Rochester, December 1981b.

Feldman, J.A. and D.H. Ballard, "Connectionist models and their properties," to appear in *Cognitive Science*, 1982.

Hayes, P.J., "Mapping input onto schemas," TR 29, Computer Science Dept., U. Rochester, 1978.

Hebb, D.O., *The Organization of Behavior*. New York: Wiley, 1949.

Hildreth, E.C., "Computing the velocity field among contours," *Proc., Motion: Representation and Perception*, ACM SIGGRAPH/SIGART Workshop, 26-32, Toronto, 1983.

Hinton, G.E., "Some demonstrations of the effects of structural descriptions in mental imagery," *Cognitive Science 3*, 1979.

Hinton, G.E., "The role of spatial working memory in shape perception," *Proc.*, Cognitive Science Conf., Berkeley, CA, 56-60, August 1981a.

Hinton, G.E., "Shape representation in parallel systems," *Proc.*, 7th IJCAI, Vancouver, B.C., 1088-1096, August 1981b.

Hinton, G.E., "A parallel computation that assigns canonical object-based frames of reference," *Proc.*, 7th IJCAI, Vancouver, B.C., 683-685, 1981c.

Iarbus, A.L. *Eye Movements and Vision*. New York: Plenum Press, 1967.

Kanade, T., "Recovery of the three-dimensional shape of an object from a single view," *Artificial Intelligence 17*, 409-460, August 1981.

Marr, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. San Francisco: W.H. Freeman, 1982.

Marr, D. and H.K. Nishihara, "Representation and recognition of the spatial organization of three-dimensional shapes," *Proc., Royal Society of London B 200*, 269-294, 1978.

McCulloch, W.S. and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics 5*, 115-137, 1943.

Palmer, S.E., "Visual perception and world knowledge: Notes on a model of sensory cognitive interaction," in D. Norman and D. Rumelhart (eds). *Explorations in Cognition*. San Francisco: Freeman, 1975.

Palmer, S.E., "Hierarchical structure in perceptual representation," *Cognitive Psychology 9*, 441-474, 1977.

Palmer, S.E., "Transformational structure and perceptual organization," *Proc.*, 3rd Annual Conf., Cognitive Science Society, Berkeley, CA, 41-49, August 1981.

Posner, M.I., *Chronometric Explorations of Mind*. Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers, 1978.

Rosenblatt, F., "The perceptron: A probabilistic model for information storage and organization in the brain," *Psych. Review 65*, 386-407, November 1958.

Schudy, R.B., "Harmonic surfaces and parametric image operators: Their use in locating the moving endocardial surface from three-dimensional cardiac ultrasound data," Ph.D. thesis, Computer Science Dept., U. Rochester, April 1982.

Shani, U., "Understanding 3-d images: The recognition of abdominal anatomy from computed axial tomograms (CAT)," TR 82, Computer Science Dept., U. Rochester, September 1981.

Shapiro, L.G., J.D. Moriarty, R.M. Haralick, and P.G. Mulgaonkar, "Matching three-dimensional objects using a relational paradigm," TR CS80014-R, Dept. of Computer Science, Virginia Polytechnic Inst. and State U., 1982.

Sloan, K.R., Jr. and D.H. Ballard, "Experience with the generalized Hough transform," *Proc.*, 5th Int'l. Conf. on Pattern Recognition, Miami Beach, 174-179, December 1980.

Stevens, K.A., "The visual interpretation of surface contours," *Artificial Intelligence 17*, 1-3, 47-73, August 1981.

Ullman, S., "Relaxation and constrained optimization by local processes," *CGIP 10*, 115-125, 1979.

Witkin, A.P. and J.M. Tenenbaum, "On the role of structure in vision," to appear, *Proc.*, 8th IJCAI, Karlsruhe, West Germany, August 1983.

Zucker, S.W., "Early orientation selection and grouping: Type I and type II processes," TR 82-6, Dept. of Electrical Engineering, McGill University, August 1982

Zucker, S.W., "Labeling lines and links: An experiment in cooperative computation," TR 80-5, Computer Vision and Graphics Lab, McGill U., February 1980.

THE USE OF DIFFERENCE FIELDS IN PROCESSING SENSOR MOTION

D. T. Lawton and J. H. Rieger

Computer and Information Science Department
University of Massachusetts
Amherst, Massachusetts 01003

Abstract

This paper develops a simple and robust procedure for recovering sensor motion parameters from image sequences induced by unconstrained sensor motion relative to a stationary environment. Difference vectors of optic flows approximate the orientations of the translational field lines in image areas in which there is depth variance between the corresponding environmental points and sufficient angular separation from the translational axis. This is developed into a procedure consisting of four steps: 1) locally computing difference vectors from an optic flow field; 2) thresholding the difference vectors; 3) minimizing the angles between the difference vector field and a set of radial field lines which correspond to a particular translational axis; and 4) extracting the translational and rotational component fields given the translational axis. This procedure does not require *a priori* knowledge about sensor motion or structure of the scene. It depends critically on sufficient variation in depth along some visual directions to endow the flow field with discontinuities. We present results of applying the procedure to sparse and low resolution displacement fields.

Presented are

Introduction

The motion of an observer/sensor is in general composed of a translation and a rotation. It generates an optic flow field in the image plane of the sensor due to changes of visual directions of details in the environment over time (Gibson *et. al.* 1955). The translation of the sensor induces a radial flow in the image with the intersection of the translational axis and image plane as its center. Sensor rotatation induces a rotational field in the image that is purely direction dependent (that is, a function of image position only).

The translational component (and its spatial and temporal derivative fields) contains, e.g., information about the shape of objects (Koenderink and van Doorn 1977), about the relative depth properties of the environment (Lee 1980, Prazdny 1980), or about motion parameters for navigating along curved trajectories (Rieger 1983). Processing optic flows induced by observer/sensor motion can be done by decomposing a flow field into its rotational and translational components and then recovering the environmental information from the translational component. Techniques for doing this generally require high resolution image displacements as input and are sensitive to the noise and error that current techniques for determining image motions typically produce. They can also involve solving complex equations and require significant computation.

The recovery of sensor motion parameters can be simplified considerably by making use of the geometrical structure of optic flows in regions corresponding to environmental depth changes. In such regions the difference vectors that have been computed over some neighborhood are oriented approximately along translational field lines. This can be seen easily for the case of details that are located exactly in the same direction from an observer/sensor but are at different depths (such as points along occluding boundaries) : as observed by Longuet-Higgins and Prazdny (1980), such points will differ in their image velocity vectors by the difference of their translational components only. This is because the rotational components of optic flows are purely direction dependent and thus equal for flow vectors positioned at the same image point. The axis of sensor translation can then be obtained from the intersection of radial fieldlines which are determined by such difference vectors. Given the axis of translation, the rotational and translational component fields are strongly overdetermined.

There are significant difficulties in applying this observation to actual image sequences. Flow fields computed from actual image sequences are not arbitrarily dense and are in fact generally sparse so there will not be two distinct flow vectors positioned at the same image point. Thus it is necessary to perform the computation using difference vectors determined from image displacement vectors which are spatially separated. From images formed at discrete, successive instants we obtain image displacements and not instantaneous optic velocities. Thus the computation must be expressed in terms of discrete sensor motions. Also, real flow fields are noisy and errorful, especially near occlusion boundaries because of the changes in image structure that occur there. Thus the procedure must be robust to such distortions in the determined difference vectors. We have found that subtracting spatially separated image displacement vectors with different corresponding environmental depths, will give reliable approximations to the correct translational field lines. Further, the resulting field of difference vectors will approximate a noisy translational displacement field which can be processed using general Hough techniques (Rieger and Lawton 1983).

Difference Vectors from Spatially Separated Flow Vectors

Here we present results on the effects of using spatially separated image velocity vectors to determine difference vectors. A difference vector formed from spatially separated image velocity vectors can be decomposed into a signal component oriented along the correct translational field line and a noise component. We find that the signal component increases for difference vectors formed at image locations where large depth changes occur in the corresponding environmental positions. It also increases with increasing distance between the difference vector and the intersection of the translational axis with the image plane. To the extent that these conditions are satisfied for an optic flow field, its difference vector field will approach the corresponding set of correct translational field lines. The computation of difference vectors over the image does not require initially determining the location of occlusion boundaries or of image areas corrsponding to large visual slant.

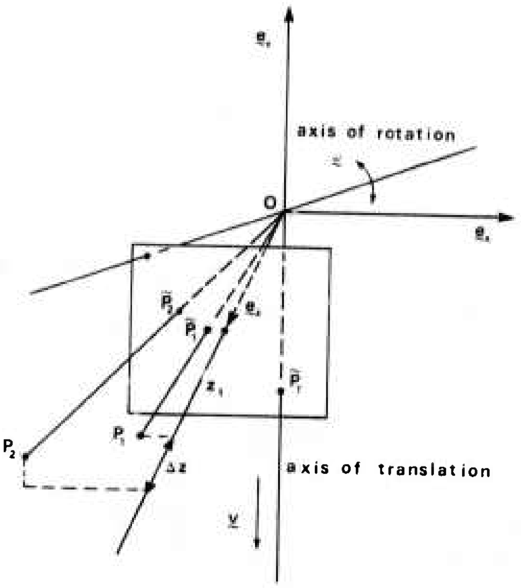


Figure 1

Consider a sensor O moving relative to a static environment. As in figure 1 the point $\tilde{P}$ at the image position $\tilde{\mathbf{r}} = (\tilde{x},\tilde{y}) = (x/z,y/z)$ corresponds to the environmental point P at the location $\mathbf{r} = (x,y,z)$. We obtain the image velocity $\mathbf{u}$ at $\tilde{P}$ by differentiating wrt time

$$\mathbf{u} = [(\dot{x} - \tilde{x}\dot{z})\,\mathbf{e}_x + (\dot{y} - \tilde{y}\dot{z})\,\mathbf{e}_y]\,/z\ .$$

Letting $\mathbf{v} = (v_x,v_y,v_z)$ and $\omega = (\omega_x,\omega_y,\omega_z)$ denote the translational and rotational velocities of O the relative motion of P becomes

$$\dot{\mathbf{r}} = -\mathbf{v} - \omega \times \mathbf{r}\ .$$

Eliminating $\dot{x}$, $\dot{y}$, and $\dot{z}$ between the above equations we can write the translational and rotational components of image velocity $\mathbf{u}$ separately

$$\mathbf{u}_T = [(\tilde{x}v_z - v_x)\,\mathbf{e}_x + (\tilde{y}v_z - v_y)\,\mathbf{e}_y]\,/z\ ,$$

$$\mathbf{u}_R = (-\omega_y + \tilde{y}\omega_z - \tilde{x}^2\omega_y + \tilde{x}\tilde{y}\omega_x)\,\mathbf{e}_x + (-\tilde{x}\omega_z + \omega_x - \tilde{x}\tilde{y}\omega_y + \tilde{y}^2\omega_x)\,\mathbf{e}_y\ .$$

Two image points $\tilde{P}_1$ and $\tilde{P}_2$ that are separated by $\tilde{\mathbf{r}}_2 - \tilde{\mathbf{r}}_1 = (d_x,d_y)$ differ in their rotational flow vectors by

$$\begin{aligned}\Delta\mathbf{u}_R &= \mathbf{u}_{R2} - \mathbf{u}_{R1}\\ &= [d_y\omega_z - d_x\,(2\tilde{x}_1 + d_x)\,\omega_y\\ &\quad + (\tilde{y}_1 d_x + \tilde{x}_1 d_y + d_x d_y)\,\omega_x]\,\mathbf{e}_x\\ &\quad + [-d_x\omega_z + d_y\,(2\tilde{y}_1 + d_y)\,\omega_x\\ &\quad - (\tilde{y}_1 d_x + \tilde{x}_1 d_y + d_x d_y)\,\omega_y]\,\mathbf{e}_y\ .\end{aligned}$$

If $\bar{r}_T = (v_x/v_z, v_y/v_z)$ denotes the intersection of the translational axis with the image plane we can rewrite the translational flow vector as $\mathbf{u}_T = v_z(\bar{r} - \bar{r}_T)/z$. Then the difference vector of two translational flow vectors at separated image positions $\bar{P}_1$ and $\bar{P}_2$ becomes

$$\Delta \mathbf{u}_T = \mathbf{u}_{T2} - \mathbf{u}_{T1}$$
$$= \frac{v_z}{z_1 + \Delta z}\left[\bar{r}_2 - \bar{r}_1 + \frac{\Delta z}{z_1}(\bar{r}_1 - \bar{r}_T)\right],$$

where $\Delta z = z_2 - z_1$ is the depth separation of the environmental details P_1 and P_2 that correspond to $\bar{P}_1$ and $\bar{P}_2$ in the image.

Now we can rewrite $\Delta \mathbf{u}$ as consisting of a component along a translational fieldline and a noise component

$$\Delta \mathbf{u} = \Delta \mathbf{u}_T + \Delta \mathbf{u}_R =$$
$$\left[\frac{v_z \Delta z}{z_1 z_2}(\bar{r}_1 - \bar{r}_T)\right]_{\text{Signal}} + \left[\frac{v_z}{z_2}(\bar{r}_2 - \bar{r}_1) + \Delta \mathbf{u}_R\right]_{\text{Noise}}.$$

For difference vectors with sufficient angular separation from the translatory axis and separation in depth $\Delta \mathbf{u}_{\text{Signal}} >> \Delta \mathbf{u}_{\text{Noise}}$.

Recovery of Motion Parameters and Depth

In order to compute difference vectors from image displacement fields formed over discrete time intervals (as opposed to continuous intantaneous image velocity fields), we have to be careful to describe all quantities with respect to the same reference system. Suppose two environmental points lie along the same ray of projection in an image at time t. Translating and rotating the sensor will displace the projections of these points to new positions in the image at time t + 1. In the image at time t + 1, the image points will be separated due to the translational component of the sensor motion (unless they are located on the translational axis). The separated image points and the intersection of the translational axis with the image plane will be collinear at time t + 1. This is the discrete analog of the fact that difference vectors at discontinuities of an instantaneous optic velocity field are oriented along translational field lines. Thus, given image displacements D1 and D2 at positions P1 and P2, the difference vector between points 1 and 2 is obtained by subtracting D2 from D1 and positioning the resulting vector at P1 + D1.

Two thresholds are used in evaluating difference vectors. The *separation threshold* determines the maximal allowable distance between displacement vectors in determining difference vectors. The *neighborhood* of a given displacement vector contains all other displacement vectors which lie within a distance determined by the separation threshold. The *length threshold* determines the minimal allowable length for a difference vector. For a given difference vector and a set of radial field lines, the *error angle* is the angle between the difference vector and the fieldline at that position.

We have found that reducing the number of difference vectors by increasing the length threshold and decreasing the separation threshold improves the fit of the difference vector field to the set of correct field lines up to a certain degree. This is because short difference vectors (compared to the local average magnitude) are more likely to deviate from the correct field lines and computing difference vectors over larger neighborhoods increases the noise components. If, however, thresholding eliminates too many difference vectors the fit detoriates, since the signal of the difference vector field becomes less distinguished for a decreasing number of vectors.

For each image displacement vector a set of difference vectors of sufficient length is determined over its neighborhood. For the resulting field of difference vectors, processing involves finding a translational axis and the corresponding set of radial field lines which minimizes the sum of the magnitude of the error angles. The procedure used is basically that used in Lawton (1982, 1983) to determine the translational axis from noisy displacement fields induced by rectilinear sensor motion. The error measure is defined on a half sphere, where points on the half sphere are possible candidates for the translational axis. The advantage of using a sphere as a domain is that it allows for a uniform, global sampling of the error function. The search process itself consists of a global sampling of the error measure to determine its rough shape using a generalized Hough transform (Ballard 1980, O' Rourke 1981) followed by a local search to find a minimum.

The computation of the sensor rotation (scaled by focal length) from the original flow field and the radial (translational) fieldlines is straightforward. Note that the components of the flow perpendicular to the radial fieldlines are induced by sensor rotation. Introducing, for convenience, a polar coordinate system (r,θ) in the image plane centered at $\bar{P}_T$ we have a system of overconstrained linear equations of the type $\mathbf{u}_R \cdot \mathbf{e}_\theta = \mathbf{u} \cdot \mathbf{e}_\theta$ in the three unknowns ω_x, ω_y, and ω_z.

Knowing the rotational parameters yields the translational and rotational component fields of the orginal flow field. The translational component is directly related to the relative depth of a scene (i.e. the depth scaled by the sensor displacement in depth δz) by the relation $z/\delta z = | \bar{r} - \bar{r}_T | / | u_T |$, where u_T is a translational flow vector in the image. If the frame rate is known the relative depth of an environmental point corresponds to its temporal separation from the sensor (under constant approach velocity). Biological systems seem to exploit this optical relation for a variety of navigational tasks (Lee 1980, Wagner 1982).

Experiments

The flow field in figure 2a shows image displacements at pixel positions having coordinates which are multiples of 8 from a 128 × 128 pixel field. The components of the displacement vectors were stored as 8 bit integers. The environment consisted of a spherical surface patch at depth of 10 units along the z axis and a background spherical surface patch at a depth of 30 units along the z axis. The obvious discontinuities in the flow field in figure 2a indicate the boundary of the nearer surface. The sensor motion consisted of an intial rotation of 0.1 radians about the (1,1,1) axis followed by a translation of 2 units along (0,0,1). The separation threshold was set to 1 pixel and the length threshold was set to 3 pixels. Figure 2b shows the average difference vectors which exceeded the length threshold. Note their occurrance along the occlusion boundary and their strong radial character. The resulting error function is shown in figure 2c (Darker in the figure corresponds to less error; also recall that this is a plot of a hemisphere in polar coordinates and not the image plane). As can be seen, it is strongly unimodal. The minimum in the global histogram corresponded to the image position (60, 60). The local search determined the minimum to be at (63, 63). The correct, subpixel, position was (63.5, 63.5). The rotational component was found by optimizing a simple expression describing the extent to which a rotational field had vector components perpendicular to the radial field lines (determined by the translational axis) which were identical to those of the orginal flow field in figure 2a. The resulting rotational and translational components are shown in figures 2d and 2e respectively. The relative depth map determined from the translational component field is shown in figure 2f encoded by intensity (darker means closer to the observer).

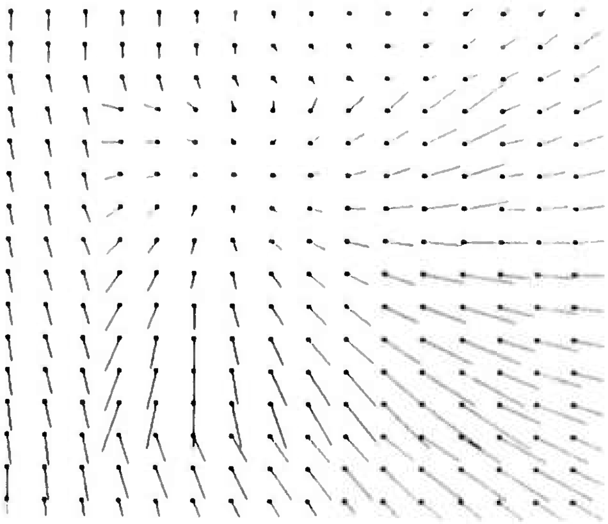


Figure 2a

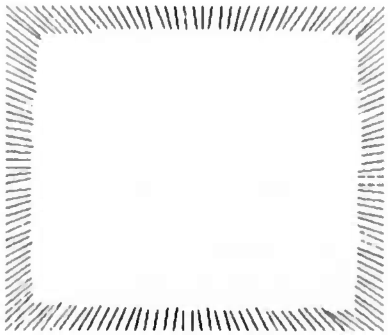


Figure 2b

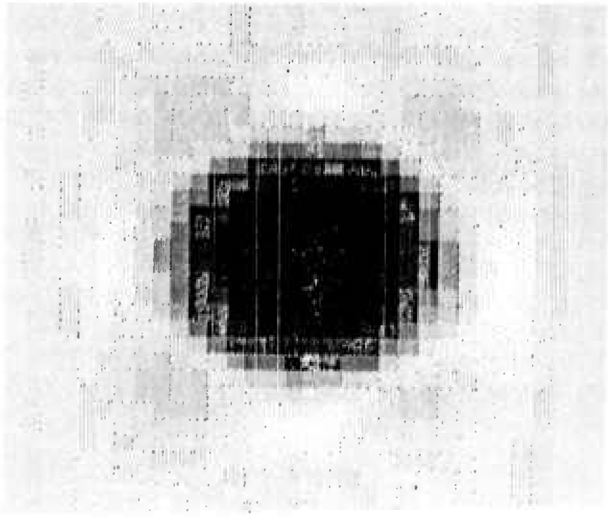


Figure 2c

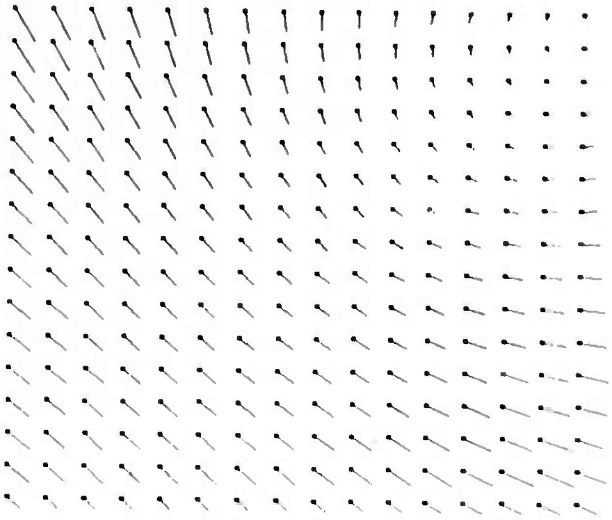


Figure 2d

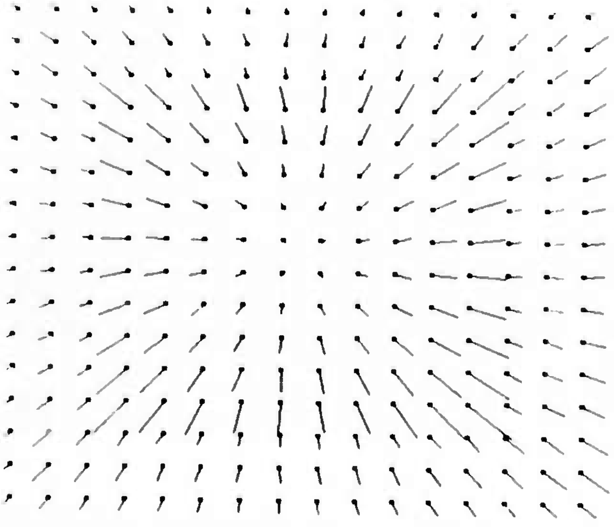


Figure 2e

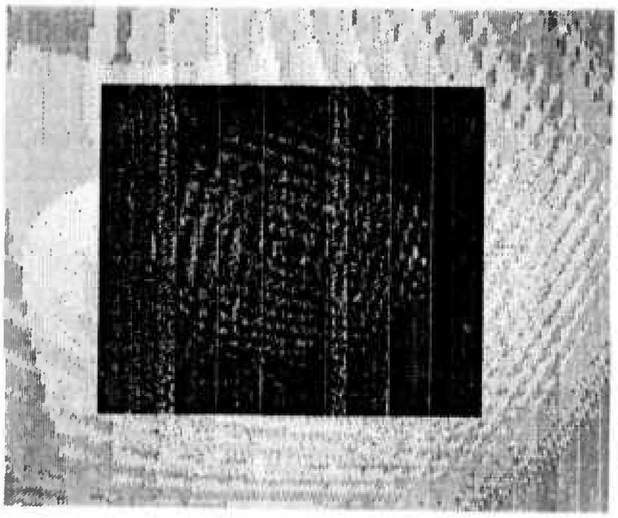


Figure 2f

The flow field in figure 3a was produced by a rotation of 0.1 radians about the x axis followed by a translation of 2 units along the z axis. The environment consisted of a planar surface oriented perpendicular to the z axis and 20 units away. This is a situation in which environmental depth variance is minimized. The difference field, computed by averaging the difference vectors in a neighborhood determined by a separation threshold of 1.0 and a length threshold of 0.0, is shown in figure 3b. The difference field in figure 3b is at a resolution 100 times greater than the field in figure 3b because the difference vectors are very small. This reflects that any inference of the translational axis in this case would be spurious.

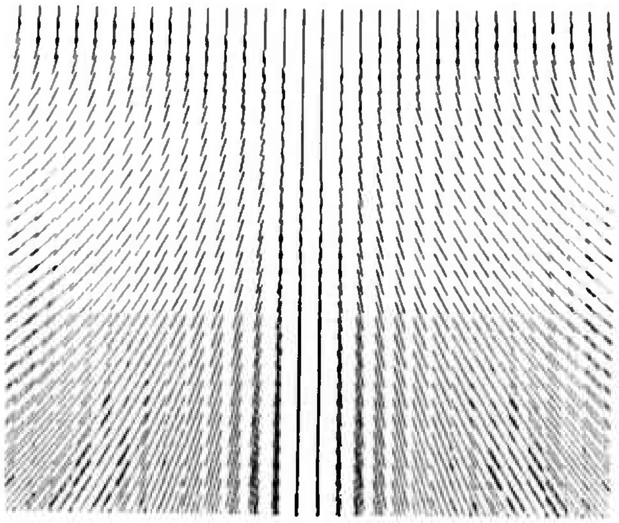


Figure 3a

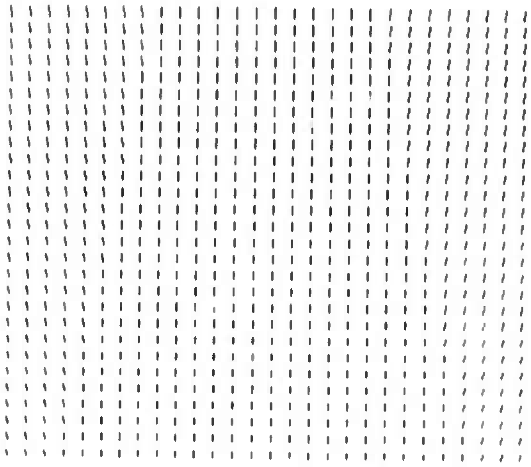


Figure 3b

The flow field in figure 4a was produced by the same motion as above except the environmental depths of the points were randomly distributed between 20 and 100 units along the z axis. The corresponding difference field is shown in figure 4b (the resolution of this figure is 3 times greater than that in figure 4a). The strong radial character of the difference field, with a Focus of Expansion at the center, is apparent.

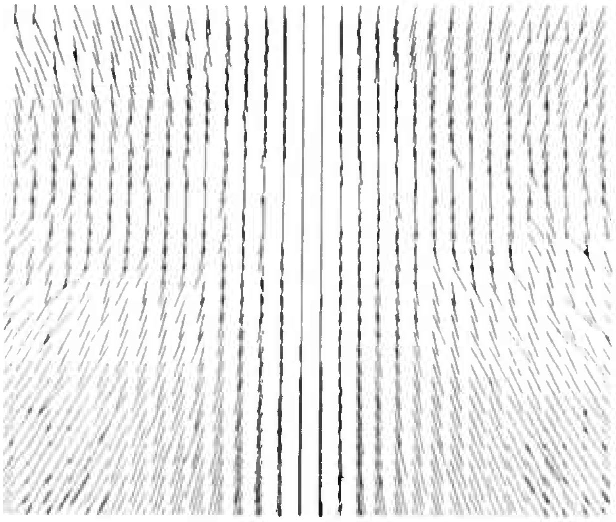


Figure 4a

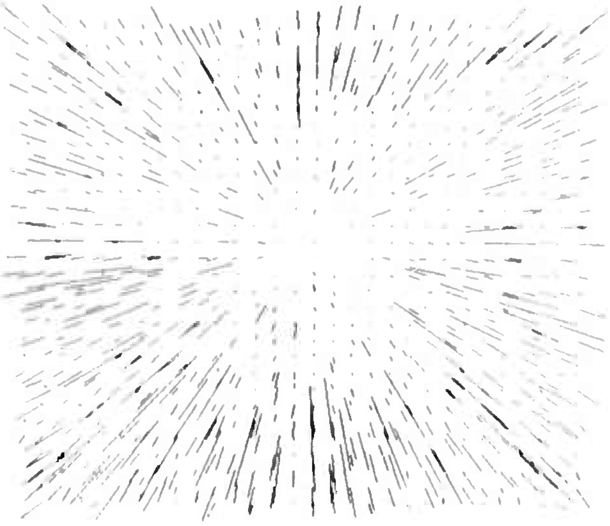


Figure 4b

Figures 5a and 5b are 128x128 pixel images with 256 intensity levels taken from a GE TN2200 solid state camera. The camera was displaced roughly in the general direction of its z-axis between two textured objects towards a textured background and then rotated about its y axis a few degrees. Figure 5c shows the displacements determined for a set of interesting points extracted from the image in figure 5a using the interest operator described in Lawton (1983). The displacements were found by correlating 5x5 pixel windows centered at these positions in the first image with 5x5 pixel windows positioned at locations within /- 15 pixels in the x and y directions in the succeeding image. Displacements for points within 10 pixels of the image boundary were ignored.

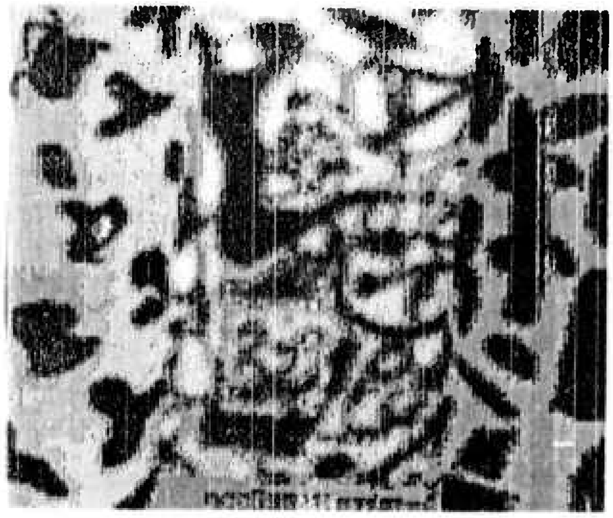


Figure 5a

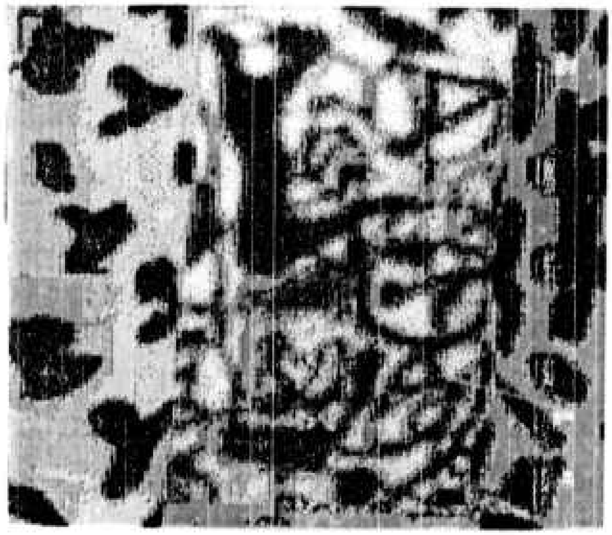


Figure 5b

The Facet Approach To Optic Flow

Robert M. Haralick and Jong Soo Lee

Departments of Computer Science and Electrical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

Abstract

The authors

The facet approach requires low level image processing techniques to he hased on fitting each local image neighborhood with a function and interpreting all processing in terms of what the processing does to this locally fit function. Using the facet approach we develop a different meaning of the usual optic flow equation. We show that it represents the intersection line of the isocontour plane with a successive image frame. The intersection line on the successive frame contains the possible match points. A unique match point can be selected hy requiring it to have the same hrightness as the given pixel. We show that this procedure amounts to assuming that all derivatives of third or higher order are negligihle and that gray tone intensity and first partial derivatives in row, column, and time must match.

They

It is shown

I Introduction

In this paper we consider the case of a camera in uniform translational motion in a static scene. In section II we derive the optic flow geometry equations for uniform translational motion and show how from the optic flow field it is possible to compute the camera velocity parameters and the depth, hoth to within an arhitrary scale factor.

Determination of the optic flow field is usually done by matching corresponding points on successive image frames. This kind of technique suffers from a potentially expensive comhinatorial complexity prohlem. In section III we apply a facet model technique to the problem of estimating the optic flow field. We show how the first order derivative optic flow equation represents the intersection line of the isocontour plane with a successive image frame. To select a unique match point on this line we require that the gray tone intensities match. We show that this procedure amounts to requiring that gray tone intensities match, and first order partials in row, column, and time match. The complexity of the technique is linear in the numher of pixels on the image. There is no comhinatorial matching. In section IV we hriefly mention some of the existing approaches for matching. In section V we present results.

II Optic Flow Geometry

Consider an image created hy a camera in constant motion, the velocity of the camera heing (a_x, a_y, a_z) in the x,y, and z directions respectively. The motion of the camera causes the position of pixels in the image to move. An image in which each pixel contains the velocity vector describing the motion of that pixel is called the optic flow image. We give a brief derivation of the optic flow.

Our perspective geometry model places the lens at the origin looking down the y-axis. The image plane is a distance of f in front of the lens. Thus a point (x,y,z) in the 3D world will have an x-position x_p on the image given by

$$x_p = f \frac{(x-a_x t)}{(y-a_y t)} \tag{1}$$

At t=0, a point (x',z') on the image corresponds to the ray $\lambda \begin{pmatrix} x' \\ f \\ z' \end{pmatrix}$ where

λ, the unknown parameter, is most directly related to the depth y of the 3D point by the relation λ = y/f. After suhstituting λx' for x and λ z' for y in equation (1) there results

$$x_p = f \frac{(\lambda x' - a_x t)}{(\lambda f - a_y t)} \tag{2}$$

The velocity u(x',z') of point (x',z') at t=0 can be ohtained as

$$\frac{\partial x_p}{\partial t} = f \frac{(\lambda f-a_y t)(-a_x)-(\lambda x' a_x t)(-a_y)}{(\lambda f-a_y t)^2} \tag{3}$$

Figure 5d shows the average difference vectors which resulted from setting the separation threshold to 10 pixels and the length threshold to 3 pixels. A plot of the error function produced using these threshold values is shown in figure 5e. The local search found a minimum at (52, 75). The correct position of the intersection of the translational axis with the image plane for the second image was determined to be at (57.97, 74.58). Since the focal length was rather long, the determined translational axis was well within 5 degrees of the actual one.

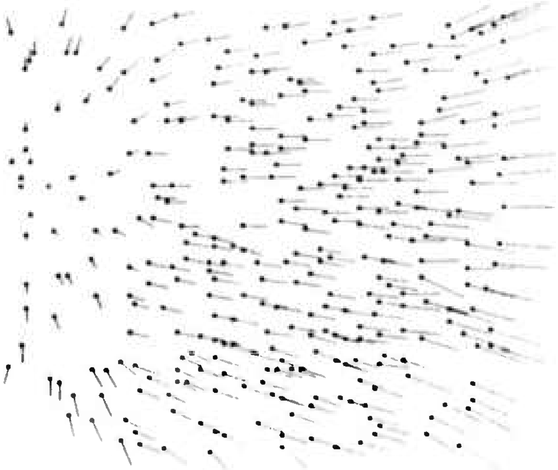


Figure 5c

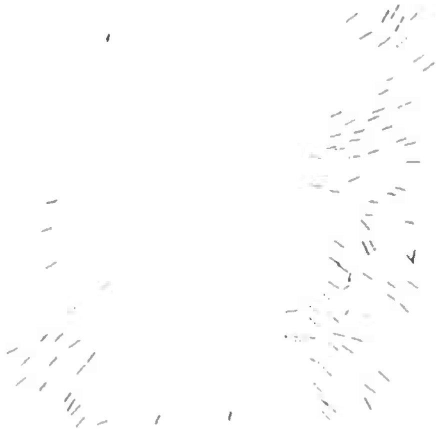


Figure 5d

This research was supported by DARPA grant N0000014-82-K-04064 to the MOTION group at UMASS.

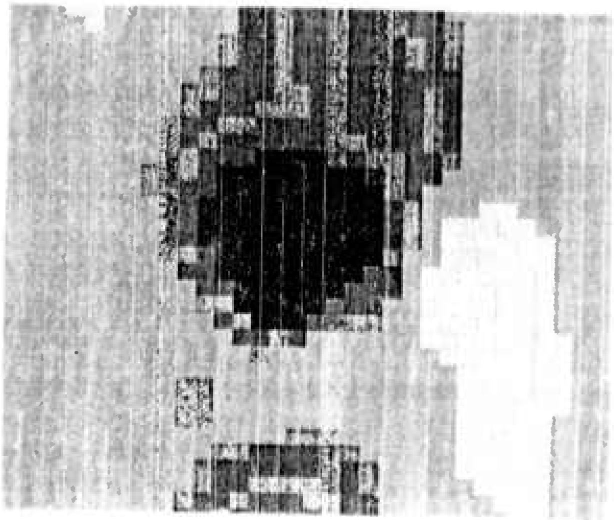


Figure 5e

References

Ballard, D. H., "Parameter Networks: Towards a Theory of Low-Level Vision", *Proc. of 7th IJCAI*, Vancouver, British Columbia, pp. 1068-1078, 1981.

Gibson, J. J., Olum, P., and Rosenblatt, F., "Parallax and Perspective During Aircraft Landings", *Am. Journ. Psychol., vol. 68*, pp. 372-385, 1955.

Koenderink, J. J., and van Doorn, A. J., "How an Ambulant Observer can Construct a Model of the Environment from the Geometrical Structure of the Visual Inflow", *Kybernetik 1977*, G. Hauske and E. Butenandt, editors, R. Oldenbourg Verlag, Munich, 1978.

Lawton, D. T., "Motion Analysis via Local Translational Processing", *IEEE Workshop on Computer Vision: Representation and Control*, pp. 59-72, 1982.

Lawton, D. T., "Processing Translational Motion Sequences", *Computer Graphics and Image Processing*, in press, 1983.

Lee, D. N., "The Optic Flow Field: the Foundation of Vision", *Phil. Trans. R. Soc. Lond. B., vol 290*, pp. 169-179, 1980.

Longuet-Higgins, H. C. and Prazdny, K., "The Interpretation of a Moving Image", *Proc. R. Soc. Lond. B., vol 208*, pp. 385-397, 1980.

O'Rourke, J., "Motion Detection Using Hough Techniques", *Proceedings of PRIP*. pp. 82-87, 1981.

Prazdny, K., "Egomotion and Relative Depth Map from Optical Flows", *Biol. Cybernet., vol. 36*, pp. 87-102, 1980.

Rieger, J. H., "Information in Optical Flows Induced by Curved Paths of Observation", *J. Opt. Soc. Am., vol. 73*, pp. 339-344, 1983.

Rieger, J. H., and Lawton, D. T., "Determining the Axis of Translation from Optic Flow Generated by Arbitrary Sensor Motion", *COINS Technical Report, 83-01*,1983.

Wagner, H., "Flow-field Variables Trigger Landing in Flies", *Nature, vol. 297*, pp. 147-148, 1982.

evaluated at t=0. Then, we have

$$u(x',z') = \left.\frac{\partial x_p}{\partial t}\right|_{t=0} = \frac{-a_x}{\lambda} + \frac{a_y}{\lambda f}x' \qquad (4)$$

The case for the z-velocity $v(x',z')$ is similar

$$v(x',z') = \left.\frac{\partial z_p}{\partial t}\right|_{t=0} = \frac{-a_z}{\lambda} + \frac{a_y}{\lambda f}z' \qquad (5)$$

For a camera in motion where $a_y \neq 0$, there will be one point (x_0,z_0) on the image whose motion will be zero. To determine this point set

$$\frac{\partial x_p}{\partial t} = \frac{\partial z_p}{\partial t} = 0 \quad \text{at } t=0 \qquad (6)$$

to obtain

$$x_0 = \frac{a_x f}{a_y} \qquad z_0 = \frac{a_z f}{a_y} \qquad (7)$$

This point is called the focus of expansion or contraction depending on whether the camera is moving toward or away from the scene.

To solve for a_x,a_y,a_z, set the parameter $\lambda(x',z')$ to an initial appropriate constant depending on scale and solve in a least square sense the system of equations:

$$\lambda(x',z')u(x',z') = -a_x + \frac{a_y}{f}x'$$

$$(8)$$

$$\lambda(x',z')v(x',z') = -a_z + \frac{a_y}{f}z'$$

This yields

$$(9)$$

$$a_y = \frac{\sum 1 \sum \lambda(x',z')[u(x',z')x'+v(x',z')z'] - \sum u(x',z')\lambda(x',z') \sum x' - \sum v(x',z')\lambda(x',z')\sum z'}{\sum (x'^2+z'^2) \sum 1 - (\sum x')^2 - (\sum z')^2}$$

$$a_x = \frac{a_y \sum x' - \sum u(x',z')\lambda(x',z')}{\sum 1}$$

$$a_z = \frac{a_y \sum z' - \sum v(x',z')\lambda(x',z')}{\sum 1}$$

where all summations are over all (x',z') in the image domain. If the scale constant for $\lambda(x',z') = k$, an unknown constant, the velocity components a_x,a_y, and a_z will all have the same multiplicative constant k. In this case, the velocity magnitude is not determined, but its direction is.

A better solution than the assumed constant λ may be obtained by iterating for reduced residual error by redefining λ to be a function of the estimated velocities.

$$\lambda(x',z') = \left[\frac{\left(-a_x + \frac{a_y x'}{f}\right)^2}{u(x',z')^2} + \frac{\left(-a_z + \frac{a_y z'}{f}\right)^2}{v(x',z')^2}\right]^{1/2} \qquad (10)$$

and then solving for a_x,a_y, and a_z in terms of the new $\lambda(x',z')$. This new λ can be substituted into equation (9) for a better estimate of the velocities. Smoothness in 3D surface can be insisted upon by taking any solution λ, considering it as an image and performing a slope facet iteration on it (Haralick and Watson, 1981).

III Calculation of Optic Flow From Image Sequence

In this section we discuss the calculation of optic flow in a time sequence of image frames and illustrate the facet approach to the optic flow computation.

Consider the case of a one dimensional sequence of frames as shown in figure 1. These frames are obviously translates of one another with a uniform motion. Instead of considering a correlation search to match each point on one frame with its corrresponding place on the next frame, consider the sequence of frames as an image each of whose rows correspond to one frame. Corresponding points on different frames have the same intensity. Thus where the one-dimensional frames are organized as an image, the corresponding points will be on equal intensity contour lines as shown on figure 2. The equal intensity contour line any point is on is easily computed as the line orthogonal to the gradient direction at that point. Thus by fitting a function to the image intensities in a local neighborhood about a point, as the facet model prescribes, and determining the gradient direction from the fit, the equal intensity contour line through the point can be determined. The match point on the next frame can be obtained without any search just as the intersection of the equal intensity contour line passing through the point with the next frame or row.

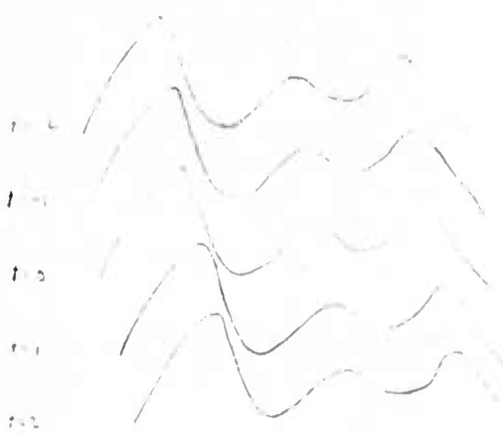
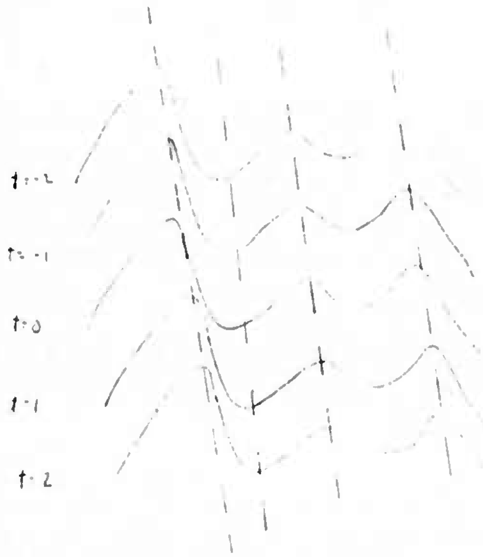


Figure 1. Illustrates a one-dimensional waveform which is translating in time.



Fignre 2. Shows the eqnal intenaity contonr lines which match correspondirg points.

In a time-varying image seqnence, the sitnation is aimilar, only the geometry is in s one dimensional higher space, a 4-dimensional space. To nnderstsnd this geometry fix attention on one pixel on one frame. Use the 3D neighhorhood (hy row, hy colamn, by image frame numher) aronnd the pixel and fit a function to the grsy tone intensities in the 3D neighhorhood. From the function fit determine the gradient vector at the given pixel position. The plane which is normsl to the gradient vector is the equal intensity contour plane passing throngh the given pixel. To determine possihle match points on the next frame intersect the eqnal intensity contour plane with the next image frame. As shown in fignre 3, the intersection is a line. The match point can he any place on this line. To determine it uniquely, find thst position on the line whose gray tone intensity is equal to the graytone intensity of the given pixel.

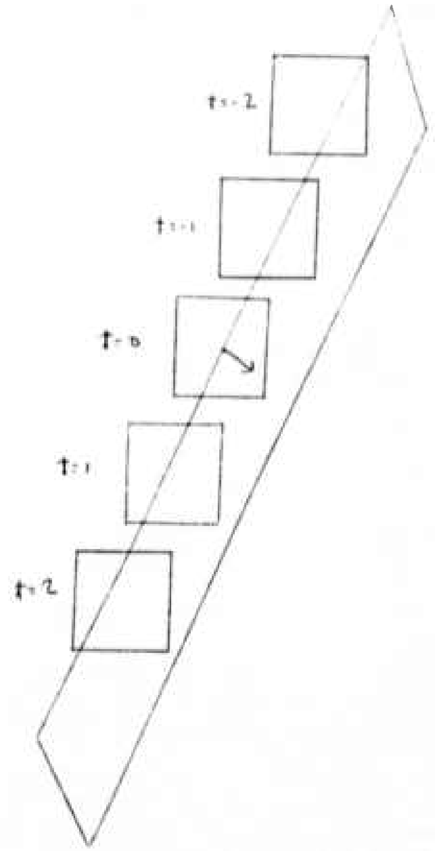


Figure 3. Shows a seqnence of fine frames, the gradient vector on frame t=0, and the plane orthogonal to the gradient vector and passing throngh the origin. The shaded area represents portions of frames to the left of the cntting plsne. The linea on frames t=-1, t=0, snd t=1 represent the intersection of the cntting plane with these frames.

III.1 Example

Conaider a local 2D neighhorhood whose gray tone intensity fqnction appears like a paraholid of the form $(r+1)^2 + (c+2)^2$ Suppoae that image frames are taken each second and that due to the camera motion the paraholid translates each successive frame by three rows and one column. Then upon fitting the gray tone intensities in a local 3D neighhorhood whose center pixel has coordinates (0,0,0) in a relative coordinate frame we determine the function

$$f(r,c,t) = (r-3t+1)^2 + (c+t+2)^2$$

Thus the parahoioid ia translating hy (3 rows,-1 column) on sncceasive frames.

The partial derivatives of f are

$$\frac{\partial f}{\partial r} = 2(r-3t+1)$$

$$\frac{\partial f}{\partial c} = 2(c+t+2)$$

$$\frac{\partial f}{\partial t} = 2(r-2t+1)(-3) + 2(c+t+2)$$

Evaluating these partials at (0,0,0) yields the gradient vector at the given pixel which is located at the origin

$$\text{grad } f = \begin{pmatrix} 2 \\ 4 \\ -2 \end{pmatrix}$$

The plane passing through (0,0,0) and orthogonal to this gradient vector is given by

$$2r + 4c - 2t = 0$$

Intersecting this plane with the next frame (t=1) produces the line

$$2r + 4c - 2 = 0$$

The gray tone intensity at (0,0,0) is given by f(0,0,0) = 5. To find the match point, find that (r,c) simultaneously satisfying the two equations

$$r + 2c - 1 = 0$$

$$f(r,c,1) = (r-2)^2 + (c+3)^2 = f(0,0,0) = 5$$

Substituting $r = 1 - 2c$ into $(r-2)^2 +_2 (c+3)^2 = 5$ yields the quadratic equation $(c+1)^2 = 0$ from which c = -1 and r = 3, the correct translation parametera.

III.2 Translational Motion

As the example suggests, the difficnlty of the computation might be in determining a real root of a polynomial. It is natural to wonder, therefore, whether it is possible to have polynomials with no real roots. We demonstrate here that for the case of translational motion, there is no possibility of the polynomial roots being only complex. To see this express the local fitted functiona f(r,c,t) as

$$f(r,c,t) = g\,(r-\alpha t,\ c-\beta t) \qquad (11)$$

explicitly indicating that the dependence between r, c, and t is constrained to translation.

The equation of the isodensity contour plane passing through (0,0) is given by

$$(r-\alpha t)\, g_r + (c-\beta t)\, g_c = 0 \qquad (12)$$

where $g_r = \frac{\partial g}{\partial r}(0,0)$

$$g_c = \frac{\partial g}{\partial c}(0,0)$$

At $t = t_0$ this plane cuts the $t = t_0$ frame producing the line

$$(r-\alpha t_0)\, g_r + (c-\beta t_0)\, g_c = 0$$

At the desired point (r,c) on this line we must satisfy the match condition

$$g(r-\alpha t_0,\ c-\beta t_0) = g(0,0)$$

Assuming g is a function for which all partial derivatives exist we may represent g(r-αt, c-βt) by its Taylor series around (0,0)

$$g(r-\alpha t, c-\beta t) = g(0,0) + (r-\alpha t_0)\, g_r + (c-\beta t_0) g_c$$

$$+ \frac{(r-\alpha t_0)^2}{2} g_{rr} + (r-\alpha t_0)(c-\beta t_0) g_{rc} + \frac{(c-\beta t_0)^2}{2} g_{cc}$$

$$+ \ldots +$$

Substituting g(r-αt, c-βt) for g(0,0) and subatituting $-(c-\beta t_0) g_c / g_r$ for $r-\alpha t_0$ yields

$$(c-\beta t_0)^2 \left(\frac{g_c^2}{g_r^2} \frac{g_{rr}}{2} - \frac{g_c g_{rc}}{g_r} + \frac{g_{cc}}{2} \right) + (c-\beta t_0)^3 [\ldots]$$

$$+ \ldots + = 0 \qquad (13)$$

Factoring ont a $(c-\beta t_0)^2$ from the left hand side and noting that the right hand aide ia zero permits us to write $(c-\beta t_0)^2 = 0$ from which we can sove for the double real root $c = \beta t_0$

III.3 Comparison

There is a relationahip between this procedure and the usual optic flow eqnation. Letting f_r, f_c, and f_t designate the partial derivatives of f with respect to r,c, and t, evaluated at the origin, the equation of the isocontour plane is given by

$$r\,f_r + c\,f_c + tf_t = 0 \qquad (14)$$

Intersecting this plane with the next image plane which is taken at t_0 seconds latter produces the line

$$-f_t = \frac{r}{t_0} f_r + \frac{c}{t_0} f_c \qquad (15)$$

Equation (15) is the usual optic flow equation (Horn and Schunk, 1980). The quantity r/t_0 represents a movement of r rows over t_0 seconds and is therefore the row velocity. Likewise c/t_0 represents the column velocity.

The difference in what we have done is that we have given equation (15) an enlarged meaning. It is the equation of a line containing the possible match points on the t_0 image frame. But since the match point must have the same brightness, we use the additional constraint that the match point (r,c) must satisfy

$$f(r,c,t_0) = f(0,0,0) \qquad (16)$$

the equal brightness constraint. This brightness constraint is used in the usual derivation of the optic flow equation so it would seem to be superfluous to use again. From our perspective we see that the isodensity contour plane is really only isodensity at the origin and as it moves away from the origin, it must be regarded as an approximation. Thus the intersection line on the successive frame is not guaranteed to have all its points be of the same brightness as the given pixel. The match condition just tells us to select that point on the line having the same brightness as the given pixel.

III.4 Why It Works

In this section we give a detailed explanation of why the procedure works. We assume that all derivatives of third or higher order are negligible and that the match conditions consist of matching gray tone intensity and gray tone first partials in row, columns, and time.

Let f with a subscript designate the corresponding partial derivative of f evaluated at r = c = t = 0. A Taylor series of f about (0,0,0) neglecting third or higher order terms is given by

$$f(r,c,t) = f(0,0,0) + rf_r + cf_c + tf_t \qquad (17)$$

$$+ \frac{r^2}{2} f_{rr} + rcf_{rc} + \frac{c^2}{2} f_{cc} + rtf_{rt} + ctf_{ct} + \frac{t^2}{2} f_{tt}$$

A pixel (r,c) having relative neighborhood coordinates on relative time image t matches pixel (0,0) on time image 0 if

(1) $f(r,c,t) = f(0,0,0)$

(2) $\frac{\partial f}{\partial r}(r,c,t) = \frac{\partial f}{\partial r}(0,0,0) = f_r$

(3) $\frac{\partial f}{\partial c}(r,c,t) = \frac{\partial f}{\partial c}(0,0,0) = f_c$

(4) $\frac{\partial f}{\partial t}(r,c,t) = \frac{\partial f}{\partial t}(0,0,0) = f_t$

Condition (1) states that the gray tone intensities must match. Condition (2) and (3) states that the gray tone spatial pattern around the original and the match pixel must match. Condition (4) states that since the motion is uniform with no acceleration the gray tone time derivatives must match.
Applying these constraints to the Taylor series we have, respectively,

$$(18)$$

$$rf_r + cf_c + tf_t + \frac{r^2}{2} f_{rr} + rcf_{rc} + \frac{c^2}{2} f_{cc} + rtf_{rt} + ctf_{ct} + \frac{t^2}{2} f_{tt} = 0$$

$$rf_{rr} + cf_{rc} + tf_{rt} = 0 \qquad (19)$$

$$rf_{rc} + cf_{cc} + tf_{ct} = 0 \qquad (20)$$

$$rf_{rt} + cf_{ct} + tf_{tt} = 0 \qquad (21)$$

Multiplying equation (19) by r, equation (20) by c, equation (21) by t and adding yields

$$r^2 f_{rr} + 2rcf_{rc} + c^2 f_{cc} + 2rtf_{rt} + 2ctf_{ct} + t^2 f_{tt} = 0 \qquad (22)$$

Substituting this back into equation (18) yields

$$rf_r + cf_c + tf_t = 0 \qquad (23)$$

the usual optic flow equation! Thus, the technique of using equation (23) and the gray tone intensity match condition (18) in essence works because it assumes that all first partials are matching. However, now we see that there need not be any problem of root finding. We just need to solve the overconstrsined system of equations

$$\begin{pmatrix} f_r & f_c \\ f_{rr} & f_{rc} \\ f_{rc} & f_{cc} \\ f_{rt} & f_{ct} \end{pmatrix} \begin{pmatrix} r \\ c \end{pmatrix} = -t \begin{pmatrix} f_t \\ f_{rt} \\ f_{ct} \\ f_{tt} \end{pmatrix} \qquad (24)$$

for the row column positior (r,c) on the specified image t.

IV Brief Matching Literature Review

Matching frsmes is an old imsge processing problem. Classically, it was solved by translating one image against the other until the correlation between the two images was highest. An equivalent calculation can be done through the use of Fourier Transform. Barnea and Silverman (1972) showed how to speed up the search by essentially not doing calculations on positions where errors must exceed the best error so far. These techniques work only for translation of one image relative to the other.

In moving images, the motion is not the same all over the image. Correlation techniques are not appropriate. Martin and Aggarwal (1979) nse bonndary information as the basis for matching. Bernard and Thompson (1980) nse a disparity analysis technique for matching. Ayala, Orton, Larson, snd Elliott (1982) use a symbolic technique for matching. Jacobus, Chien, and Selander (1980) use a graph matching technique. Aggarwal, Davis, and Martin (1981) review techniques for establishing corresponding points on images. The problem with most of these techniques is that they must employ some combinatorial computation to establish the match. This kind of compntation is very expensive.

Techniques which do not involve combinatorial matching inclnde Limb and Murphy (1975) who relate image intensity changes over time to spatial gradient and Fennema and Thompson (1979) who nse a gradient intensity transform method. Both these techniques are similar to the one presented in this paper in that they establish the match using only local neighborhood analysis.

V Resnlts

To confirm that our theory works, we tested onr algorithm on 3 kinds of image sequences. The image sequences describe the movement of an ellipsoid in translation, magnification, and rotation. The time interval between two consecntive images in a sequence corresponds to one pixel difference in an image.

To compute an optic flow vector of a pixel on the image at t=0, we determined the nnderlying fnnction over its 3-D neighborhood nsing a 3-D cubic discrete orthogonal polynomial basis, and, next, derived the 4 constraining eqnations (Eq24) on the row and column components of the optic flow vector at the center of the pixel. To solve the over-constrained equations, we, first, obtained two singular values using the Singular Value Decomposition routine of the Linpack, and, next, determined the least square solution from the singular values.

To the time sequence of an ellipsoid moving with the velocity of r/t=1 and c/t=-.8 shown in Fig 4, we applied the above method with the 3-D neighborhood (5x5x5) and obtained the optic flow image shown in Fig 5. At the pixels on or near the boundsry cf the ellipsoid, the optic flow vector obtained does not show the correct movement of the ellipsoid because neighborhoods contain a mixture of stationary background, moving ellipsoid, thereby providing inconsistent information for fitting. The reason for the inconsistency is that the center pixel may be in the stationary bsckground but it has neighbors which are not. These neighbors generate an estimated surfsce which are not. These neighbors generste an estimated surface which has some curvature for the center pixel.

To reject an optic flow vector obtained from such a neighborhood, we compute the ratio of two principal curvatures from the underlying gray tone intensity surface determined at t=0. From the histogram of this curvature ratio over all the neighborhoods in the image sequence shown in Fig 6, we can determine a threshold value for the ratio at about 0.05. Fig 7 illustrates the result. The pixels which still hsve incorrect directions correspond to neighborhoods with large fitting errors over the center pixel. Fig 8 illnstrates a histogram of the center pixel fitting error. Thresholding the original optic flow image with the cnrvature ratio of .05 and rejecting the optic flow vectors obtained from the underlying function having fitting error of more than 1, we have the optic flow image shown in Fig 9. Rejecting the vectors having fitting error of more than .01, we have the optic flow image shown in Fig 10.

For the time sequence of the ellipsoid moving backwards with the magnification factor .95 shown in Fig 11, we obtain the optic flow image shown in Fig 12 where we thresholded the original optic flow image with the ratio .05 and rejected the vectors obtained from underlying fnnction having fitting error of more than 1. In the same way, for the time sequence of the ellipsoid rotating clockwise with the angnlar velocity .1 radian shown in Fig 13, we obtain the optic flow image shown in Fig 14.

References

Aggarwal, J. K., Davis, L. S., Martin, W. N., Correspondence Processes in Dynamic Scene Analysis, Proceedings of the IEEE, Vol. 69, No.5, May 1981, pp. 562-572.

Ayala, I. L., Moving Target Tracking Using Symbolic Registration, IEEE Transactions On PAMI, Vol. PAMI4, No.5, September 1982, pp. 515-520.

Barnard, Stephen T., Thompson, William B., Disparity Analysis of Images, IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. PAMI-2, No.4, July 1980, pp. 333-340.

Barnea, D. I., Silverman, H. F., A Class of Algorithum For Fast Image Registration, IEEE Transactions On Computers, Vol. C-21, No.2, 1972, pp. 179-186.

Fennema, Claude L., Thompson, William B., Velocity Determination in Scenes Containing Several Moving Objects, Computer Graphics And Image Processing, Vol. 9, (1979), pp. 301-315.

Horn, B. K. P., Schunck, B. G., Determining Optical Flow, Artificial Intelligence, No.17, (1981), pp. 185-203.

Jacoubs, Charles J., Chien, Robert T., Motion Detection and Analysis of Matching Graphs of Intermediate-Level Primitives, IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. PAMI-2, No.6, November 1980, pp. 495-510.

Limb, J. O., Murphy, J. A., Estimating the Velocity of Moving Images in Television Signals, Computer Graphics And Image Processing, Vol. 4, 1975, pp. 311-327.

Martin, W. N., Aggarwal, J. K., Computer Analysis of Dynamic Scenes Containing Curvilinear Figures, Pattern Recognition, Vol. 11, December 1978, pp. 169-178.

Acknowledgement

This research has been supported by Lockheed Palo Alto Research Laboratory

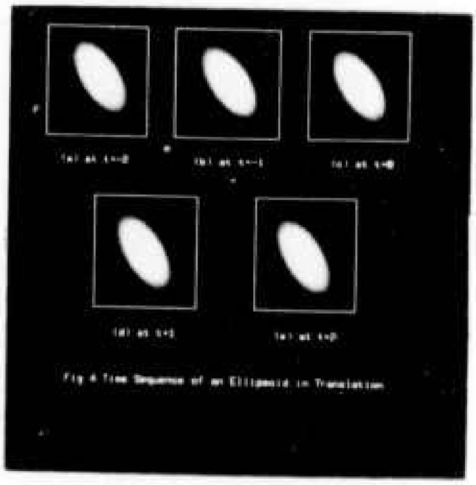
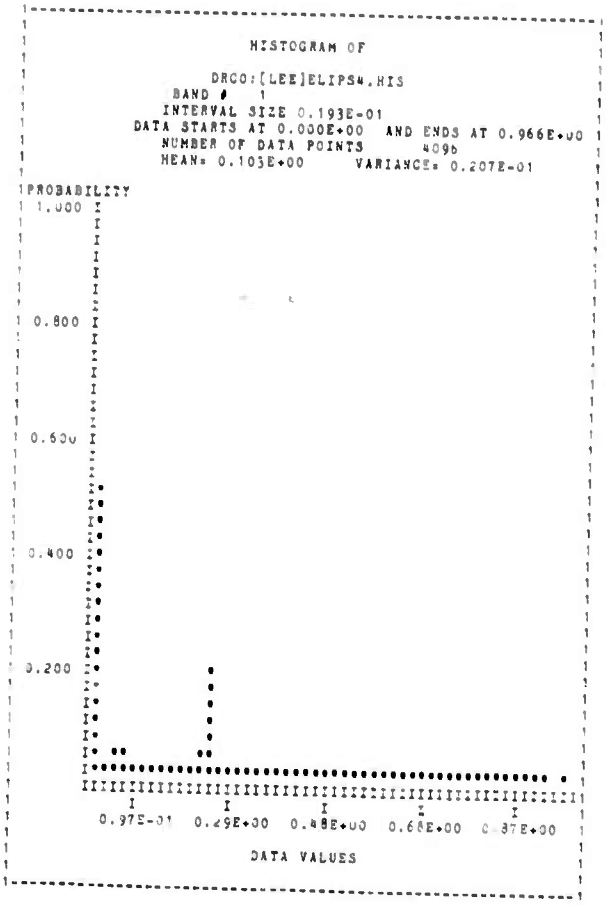


Fig 4 Time sequence of an ellipsoid in translation



```
HISTOGRAM OF

DRCO:[LEE]ELIPS4.HIS
BAND #    1
INTERVAL SIZE 0.193E-01
DATA STARTS AT 0.000E+00  AND ENDS AT 0.966E+00
NUMBER OF DATA POINTS        4096
MEAN= 0.103E+00      VARIANCE= 0.207E-01
```

Fig 6 Histogram of curvature ratio

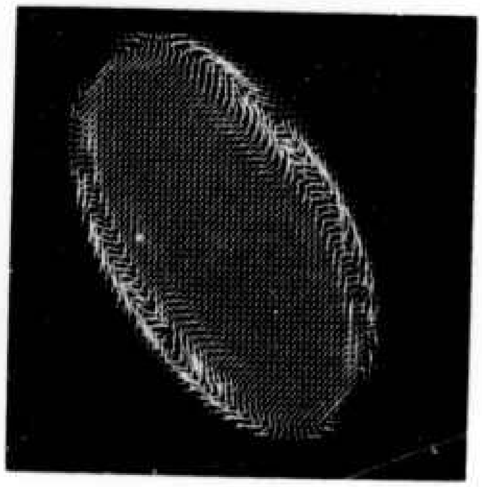


Fig 5 Original optic flow image

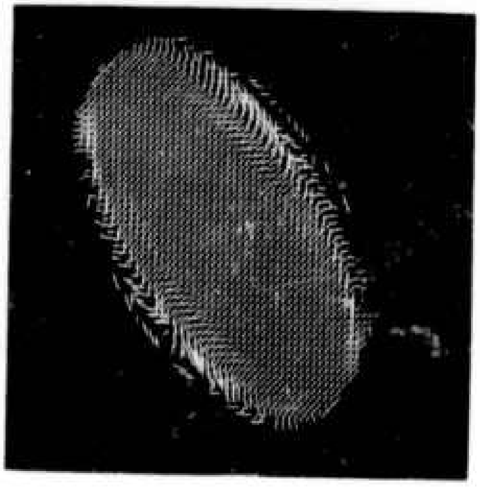
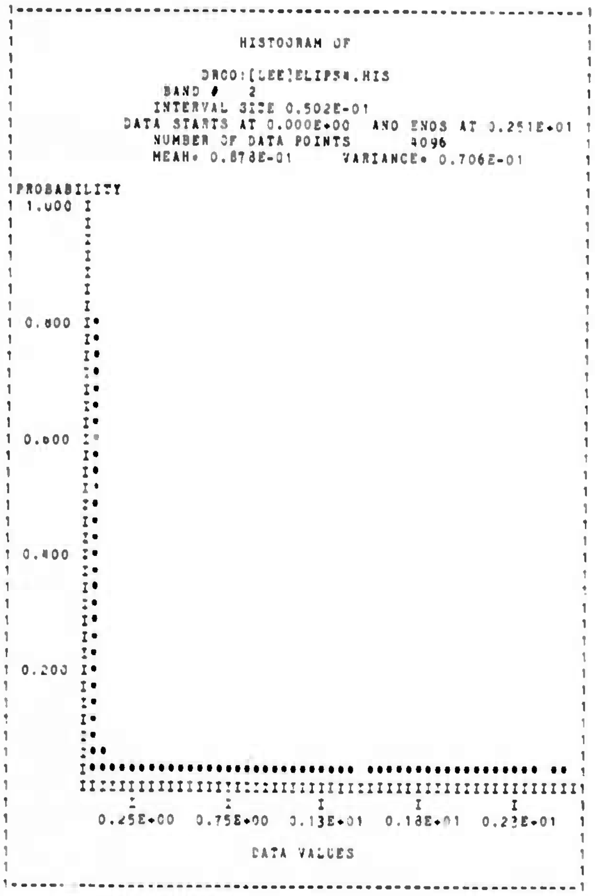


Fig 7 Thresholded optic flow image



```
                    HISTOGRAM OF

                 DRCO:[LEE]ELIPS4.HIS
            BAND #     2
           INTERVAL SIZE 0.502E-01
        DATA STARTS AT 0.000E+00   AND ENDS AT 0.251E+01
           NUMBER OF DATA POINTS        4096
           MEAN= 0.878E-01      VARIANCE= 0.706E-01
```

Fig 8 Histogram of fitting error

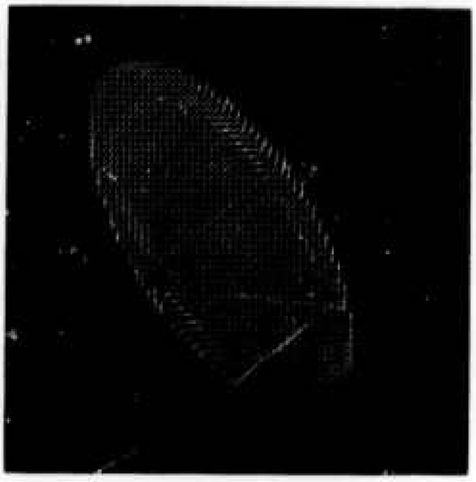


Fig 9 Optic flow image without rejected vectors

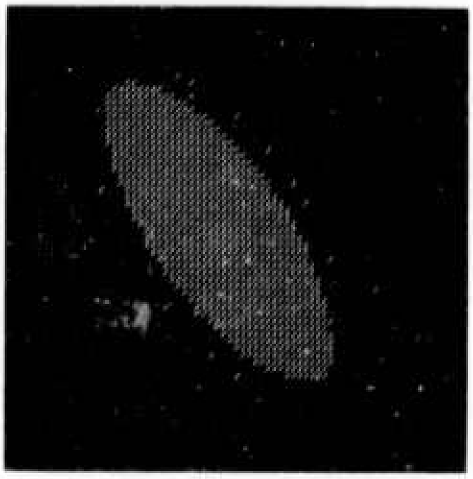


Fig 10 Optic flow image

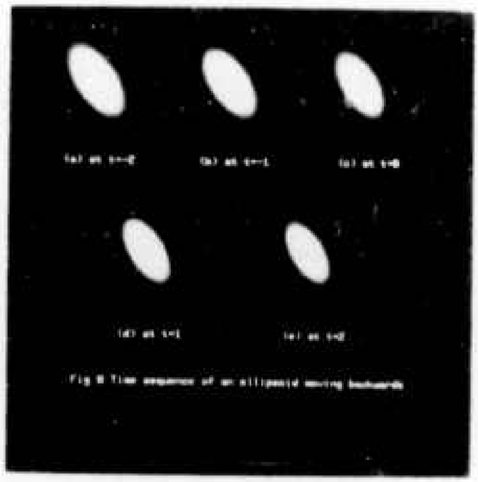


Fig 11 Time sequence of an ellipsoid moving backwards

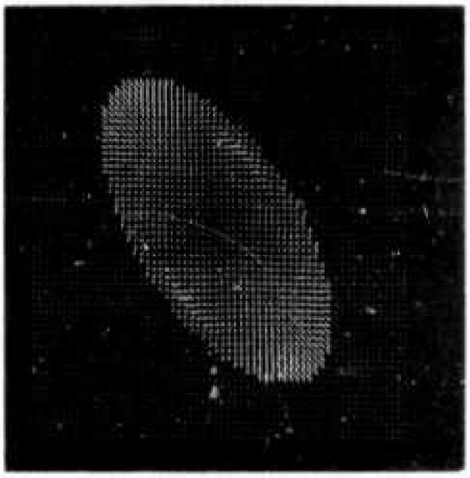


Fig 12 Optic flow image obtained from Fig 11

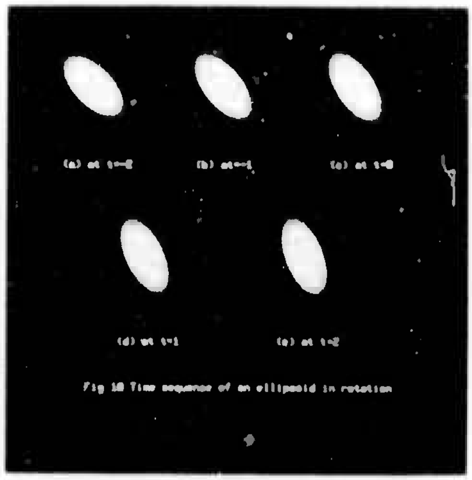


Fig 13 Time sequence of an ellipsoid in rotation

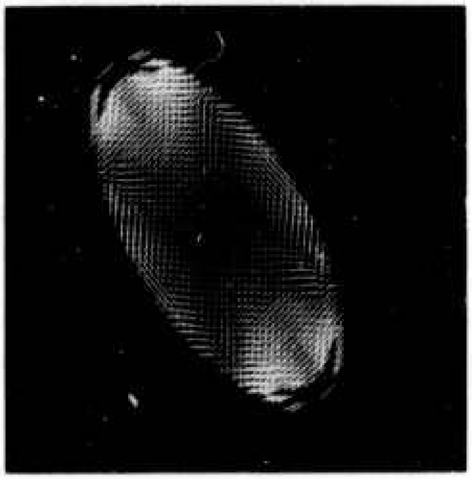


Fig 14 Optic flow image obtained from Fig 13

SPECIAL PURPOSE AUTOMATIC PROGRAMMING FOR 3D MODEL-BASED VISION

Chris Goad
Department of Computer Science
Stanford University,Stanford Ca 94305

Three dimensional

Abstract

A method for the automatic construction of fast special purpose vision programs is described. The starting point for the automatic construction process is a description of a particular 3D object. The result is a fast special purpose program for recognizing and locating that object in images, without restriction on the orientation of the object in space. The method has been implemented and tested on a variety of images with good results. Some of the tests involved images in which the target objects appear in a jumbled pile. The current implementation is not fully optimized for speed. However, evidence is given that image analysis times on the order of a second or less can be obtained for typical industrial recognition tasks. (This time estimate excludes edge finding).

1. Introduction

In many practical applications of automated vision, the vision task takes the form of recognizing and locating a particular three dimensional object in a digitized image. The exact shape of the object to be perceived is known in advance; the purpose of the act of perception is only to determine its position and orientation relative to the viewer. This is model based vision in its strict form.

Most industrial applications of vision have this property, and also the property that the same object (or, more precisely, objects of the same shape), must be located in many images. In this kind of situation, it is desirable to split the computation into two stages: an analysis or precomputation stage, in which useful information about the (unchanging) object is compiled, and an execution or runtime stage, in which this information is exploited for the rapid recognition of the object in an image. The reason for breaking up the computation in this way is of course that the analysis only needs to be done once, whereas its results can be exploited repeatedly. Bolles[Bolles 1982] among others, has taken this general approach to the model based vision problem.

The advance analysis stage may take a variety of forms. In our work, this stage involves a kind of automatic programming. A description of the object to be recognized is "compiled" into a special purpose program whose only function is to recognize that one object in digitized images. In the second, runtime stage, individual images are processed by the special purpose program produced at the first stage.

This formulation of the work accomplished by the advance analysis is very unrestrictive. It makes no commitment as to the algorithm which is used to process images; rather the algorithm may be chosen according to the object which is to be recognized. The problem of finding the best algorithm among all algorithms for a given object is intractable. However, we may attempt to construct special purpose algorithms for object recognition within a restricted class of algorithms, and hope for good, though not optimal, results.

In this paper, we describe a method for automatically constructing special purpose programs for 3D object recognition. To be precise, we mean by 3D object recognition the recognition of three dimensional objects in ordinary light intensity images (not, eg, range images), where no restriction is made on the orientation of the object with respect to the camera. Although we speak of recognition, the process of recognition delivers information not only about the presence or absence of the object in the image, but also the position and orientation of the object if it is present. The method relies on matching object features to image edges. We will not concern ourselves here with how image edges are extracted from pixel data. The method does not rely on perfect results from the edge finder (if it did, it would be of no practical interest). The special purpose programs generated are quite fast. To give a very rough idea of how fast, our method should allow the recognition of ordinary industrial objects in moderately complex background in a second or less on a 1 MIP computer; this is the time required for the matching process, and excludes the time required for edge finding. The data on which this kind of general speed estimate is based will be given later in the paper.

2. A general strategy for special purpose automatic programming

The general features of the model based vision problem which make it a candidate for special purpose automatic programming are shared by a wide variety of computational problems. The features in question are that the inputs to the computation are delivered in two stages, and that for each first stage input (here, the object to be recognized), many second stage inputs (here, images) must be treated. The special purpose automatic programming problem in its general form can be stated as follows.

Let us suppose that a function f of two inputs x and y must be computed repeatedly under conditions where many values of y must be treated for each value of x. Then we attempt to devise, for each x, a special purpose program P_x with $P_x(y) = f(x,y)$. More precisely, what is wanted is an automatic process for constructing the programs P_x - that is, synthesis method M with $M(x) = P_x$.

The informal strategy which we use for the model based vision problem can also be described at this level of generality. The strategy is just that of starting with a general purpose program for doing the computation in one stage, and using specialized variants of this program as the templates from which special purpose programs are developed. Suppose, again, that $f(x,y)$ must be computed repeatedly with slowly changing x and rapidly changing y. Suppose also that we have in hand a program $G(x,y)$ which computes $f(x,y)$ in one stage. We begin by "unwinding" G as it applies to the value x for which a special purpose program P_x is wanted; this unwound program $G'_x(y)$ is then optimized to get the desired result. The unwinding is done by a kind of symbolic execution, in which loops are unwound when possible, and recursions are unfolded. If the procedures for unwinding and optimization are mechanical in nature, then together they constitute what we have called a synthesis method for f. The unwinding and optimization procedures need not be applicable to arbitrary programs; they can be custom designed for the particular program G at hand.

In general, the program G does not have to be completely specified - G may itself be a template for an algorithm, with many details left out. The details may be filled in after - rather than before - G has been unwound.

In our work on vision, we begin with a one stage algorithm template $G(x,y)$ which takes an object description x, and an image y, and identifies instances of x in y. G is a template for an algorithm in the sense of the last paragraph; many of the details of its operation will be specified only after it has been unwound for particular objects.

Model-based vision is the second problem to which we have applied this style of special purpose automatic programming. [Goad 82] describes the earlier application to hidden surface elimination in 3D computer graphics, and also contains a general discussion of special purpose automatic programming as it relates to other work on the automatic construction and manipulation of programs.

3. A one stage algorithm for model based vision

The one stage algorithm $G(x,y)$ which we start with is a simple sequential matching procedure. The kind of object description expected by G is a list of object features, together with conditions on their visibility. For the current purposes, an object feature is taken to be a curve along the object surface at which either a surface normal or a reflectivity discontinuity occurs. We will restrict ourselves to straight line segments rather than considering arbitrary curves. So, informally, an object feature is just a straight edge on the object. For each object feature, G also needs to know the range positions in space from which that feature is visible (means for representing this information will be described later). There is no need for the list of features making up an object description to be exhaustive; it is sufficient that enough features be included to make reliable recognition possible. As a result, the kind of description of an object which is needed for its recognition is much less extensive than that needed for displaying it.

The image description expected by G is of the same general kind as the object description: it is a list of features. In particular, the image features employed by G are of exactly the kind which the object features give rise to in the imaging process: they are straight segments in the image along which an intensity discontinuity occurs. The process by which this kind of image description is generated from raw pixel data will not be discussed in this paper. In our experiments, we used an edge detection program written by David Marimont[Marimont 1982] The program convolves the image with a lateral inhibition operator, detects zero-crossings in the laterally inhibited image, and then performs linking. Straight edges are arrived at by applying a simple segmentation scheme (we added this last step to Marimont's algorithm).

Although we will restrict ourselves in this paper to treating edge features, the same methods would apply to any kind of object feature which gives rise in a predictable way to an image feature.

The operation of G may be described in general terms as follows. G performs a simple depth-first search for a match between object and image edges. At any given time in the search, G's state includes a currently hypothesized match M, and a current hypothesis about the position and orientation of the object relative to the camera. The hypothesis concerning the location of the object gives bounds on the location parameters, and not exact values. In the main loop of the algorithm, G attempts to extend and refine its current hypothesis by means of the following three steps (which lie at the heart of many algorithms for perception):

(1) Predict: An object edge is selected which has not yet been matched by any image feature. Based on the current hypothesis, the position and orientation of its projection in the image is predicted.

(2) Observe: The list of image edges is checked to see whether any has the predicted qualities.

(3) Back-project: If an edge with predicted qualities was found in step (2), then extend the match to include this edge, and use the measured position and orientation of the edge to refine the current hypothesis as to the location of the camera.

The algorithm repeats this loop until either a satisfactory match is found, or until the algorithm fails to observe a predicted edge. In the latter case, the algorithm backtracks to the last choice point. Choice points arise when more than one image edge appears in a predicted position and orientation.

This is a sketch of the algorithm. Before supplying further details, some definitions are needed.

It will be convenient to work in object centered coordinates: The object will be thought of as fixed, and the position and orientation of the viewer as the unknown to be determined. An object edge is always regarded as an oriented segment (given by the *ordered* rather than unordered pair of its end points), while an image edge may or may not be oriented. The imaging process is given by the ordinary perspective transformation. The parameters of this transformation which derive from the camera model are the distance from the point of projection to the image plane, and the field of view, given by a rectangle on the image plane. Assume that these parameters are fixed. Let p be a 3D position, q a 3D orientation, and x an object edge. Then $\Downarrow([p,q],x)$ denotes the oriented image edge which results from viewing x from camera position and orientation $[p,q]$. The edge x may not be visible from $[p,q]$, either because x is occluded, lies on the "wrong side" of the object, or because the projection of x onto the image plane lies outside of the field of view. In these cases, $\Downarrow([p,q],x)$ is undefined. We will write $\Downarrow(x)$ to denote the projection of x when the parameters $[p,q]$ of the projection are clear from context. At this point we make several assumptions about the imaging geometry.

First we assume that, in the images which are to be analyzed, the object sought is either not visible at all, or lies entirely within the field of view. Second, we assume that the field of view is sufficiently narrow that changes of the orientation parameter q at a fixed position p have only negligible effects on the lengths of projected edges, and the angles between them. Thus, while a change in q will in general cause some of the image to move out of the field of view, that part of the image which remains visible will have undergone only a 2D rotation and translation - to within a small tolerance. This criterion is met in typical industrial imaging situations. Finally, we make the more restrictive assumption that the distance from the camera to the object - or more precisely from the perspective projection point to the origin of the object centered coordinate system - is known in advance. (This restriction is made to simplify the exposition, and does not apply to the implementation described later). Thus the position parameter p is restricted to lie on a sphere about the origin. Without loss of generality, we may assume that this is the unit sphere.

We will refer to a set of positions on the unit sphere as a *locus*. A locus is to be thought of as a set of possible camera positions. Let X be an object description and Y an image description. Recall that X is a list of edges together with visibility conditions. The visibility condition for each edge e in X is given by the locus of points from which that edge is wholly visible. This is called the *visibility locus* of e. (Methods for representing such loci will be given later). Now, a *match* M between object edges X and image edges Y is an assignment of image edges to object edges. A match also assigns orientations to the otherwise unoriented image edges. For any object edge, $M(e)$ denotes the oriented image edge (if any) assigned to it by M. The assignment may be partial; that is, for some e, $M(e)$ may be undefined.

A match M is consistent with a camera position and orientation $[p,q]$ if for each object edge e, the projection $\Downarrow([p,q],e) \approx M(e)$ to within errors in measurement. A match M is consistent with a camera position p if there is some orientation q such that M is consistent with $[p,q]$. A match is consistent with a locus L if it is consistent with every position in the locus.

As indicated earlier, the algorithm G conducts its search for a match by attempting at each point to extend and refine its *current hypothesis* about the imaging situation. This hypothesis has two parts: the match M found so far, and the locus L of possible positions of the camera. In the course of the matching process, the consistency of L with M is maintained (modulo errors in measurement).

Now we can be more explicit about how the basic predict-observe-back-project loop is carried out. We may restrict ourselves to considering the case where at least one edge has already been matched, since prediction and back-projection do not apply to the matching of the first edge; all acceptable candidates for matches to the first edge must be considered, wherever they appear in the image. Let M be the current match, and L the current locus. G selects an object edge e which has not as yet been matched. For the sake of brevity, when we refer to the "position" of an edge, we will henceforth mean its position *and* orientation. Bounds on the position of the image $\Downarrow(e)$ of the new edge e can be predicted simply by selecting an already matched edge e_0, and computing the bounds on the possible position of $\Downarrow([p,q],e)$ relative to $\Downarrow([p,q],e_0)$ as p ranges over the current locus L (recall that the value of q does not affect relative measurements). This prediction, together with the known position of $M(e_0)$, give predicted bounds on the position of the image $\Downarrow(e)$ of e.

A similar method can be used for back-projection. Suppose that an image edge $M(e)$ has been matched to the object edge edge e. Back-projection consists of restricting the current locus L to the smaller locus L'

which is consistent with the measured position of $M(e)$. Let e_0 be some already matched edge, and $M(e_0)$ its match in the image. L' is then just the set of camera positions p in L from which the predicted position of $\Downarrow([p,q],e)$ relative to $\Downarrow([p,q],e_0)$ is the same as the measured position of $M(e)$ relative to $M(e_0)$, to within measurement error.

This scheme preserves consistency of matches if measurement errors are negligible.

The algorithm G as described so far does not take into account the fact that any given object edge may or may not be visible depending on the position of the camera. The following extension to G deals with this aspect of the matching problem.

In the prediction step, rather than selecting an arbitrary unmatched edge as was done before, G selects an edge e whose visibility is consistent with the current hypothesis (formally, and edge whose locus of visibility intersects the currently hypothesized locus). Then, a case analysis according to whether the edge is actually visible is performed.

On one side of the case analysis, G assumes that e is visible, and restricts the currently hypothesized locus accordingly: the new restricted locus is the intersection of the current locus with the locus of visibility of the edge. Then G proceeds as before: it predicts the position of e, looks for it in the predicted position, and back-projects if found.

On the other side of the case analysis, G assumes that the edge is invisible, and again restricts the currently hypothesized locus accordingly: this time the restricted locus is the intersection of the current locus with the complement of the locus of visibility of e. If the restricted locus is empty, then the current attempt at a match has failed, and G backtracks to the last choice point. If the restricted locus is non-empty, G proceeds by selecting another object edge to match.

This case analysis step constitutes a choice point for back-tracking. Thus, for each edge selected for matching, G assumes first that the edge is visible and looks for it. If this course of action leads to a good match, then all is well. Otherwise G backtracks and looks for a match under the assumption that the edge is invisible.

In the above, an edge should be considered visible only if - in addition to meeting the usual criteria - its projection is long enough to allow detection by the edge finding program. An edge which presents itself end-on to a given camera position should not be considered visible from that camera position.

Among other details about G which have been suppressed so far is the method by which loci of camera positions are represented. A very simple representation is adequate for our purposes. Suppose that we have a scheme for partitioning the unit sphere into an arbitrary number patches such that the diameters of the patches go to zero as their number increases. Then a locus can be represented to any desired resolution by a set of patches from a partition of adequate size. More precisely, a locus L is to be represented by the set of patches from the partition which contain some point of L. The resolution of this representation is bounded by the maximum diameter of a patch. Thus loci are represented by subsets of a finite set. These in turn may be represented by bit maps: one bit is allocated to each patch on the sphere. Bit maps are a particularly good representation for the current application, since the operation most frequently performed on loci is intersection, and intersection of bit maps is very fast on any computer. The particular scheme which we have chosen for partitioning the sphere is not the best but the simplest. The partition is generated by first imposing a regular grid on the faces of a cube. The cube is then projected radially onto the sphere. The patches on the sphere which we end up with are simply the projections of grid elements from the faces of the cube. In recent experiments, we have used 6 by 6 grids on the faces of the cube, yielding a total of 218 patches. This representation is depicted in figure 1. One 36 bit PDP-10 machine word is allocated to each face. So, a locus is represented by 6 machine words.

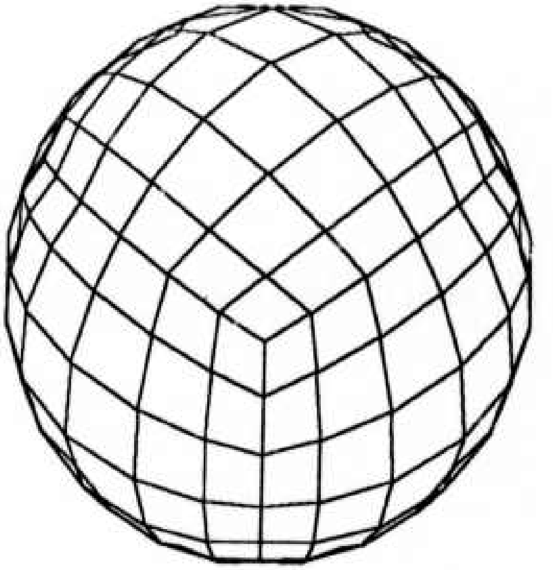


figure 1

We are still not done with the development of G. A major shortcoming of G as described so far is that it relies on perfect performance by the edge finder - each edge on the object which is in view must be detected by edge finder if the method to function properly. This kind of perfect performance is not obtained by existing edge detection programs, nor can it be obtained by any edge detector which relies on local image intensity discontinuities to detect edges, since object edges do not always give rise to such intensity discontinuities.

The matching algorithm may be modified in order to take into account the imperfections of the edge finder by accepting matches in which only a fraction of the expected edges are present. If such a modification is made, the criteria of match success and match failure become more complicated. It is necessary to determine conditions under which a partial match should be dropped because of inadequate success in finding predicted edges, and also conditions under which a match should be accepted as reliable evidence that the object being looked for has actually been found.

In order to do this, we would like to be able to assess the probability that a given partial match arose from the object in the manner claimed. If this probability is very low, then the match should be dropped, and if very high, it should be accepted. For intermediate values, more edges should be matched, if possible, in order to accumulate further evidence.

Direct estimates of this probability are difficult to make in the usual cases. Nonetheless, we can proceed by estimating conditional probabilities, and use qualitative considerations to get from these conditional probabilities to the needed conclusions concerning a given match.

On the one hand, it is possible to estimate the probability that the configuration of edges making up a match arose by chance given any particular assumed "background" distribution of edges giving rise to chance matches, since the specificity of each prediction involved in the match is known, as is the number of such predictions which have been met. We will refer to the inverse of this probability as the "reliability" of the match. The background distribution of edges usually cannot be derived from first principles, but is best determined by gathering statistics on sample images of the kind on which the algorithm is to be used.

On the other hand, suppose that we have a partial match in which some fraction of the predicted edges are missing. Then we can estimate the probability that the given set of edges would be missed by the edge detector under the assumption that the partial match did arise as claimed. We will refer to this probability as the "plausibility" of the match. Estimating plausibility requires information about the performance of the edge detecter. As in the case of edge distributions, deriving this kind of information from first principles is difficult; again, compiling statistics from sample images is a better idea. Note that underestimating the performance of the edge detector will lead to robust performance by the matching algorithm.

So the reliability of a match measures how unlikely it is to have arisen assuming it is in fact incorrect, and its plausibility measures how likely it is to have arisen assuming it is in fact correct. Assuming that the presence of the object in the field of view is moderately likely and there are unlikely to be impostors of the object in view, it follows from Bayes' Rule that high reliability provides strong evidence that the match is correct, while very low plausibility provides strong evidence that the match is incorrect. (Notes: (a) Low reliability does not provide evidence that the match is incorrect, nor does high plausibility provide evidence that the match is correct (b) By an "impostor" in the above, we mean an object which is regarded as distinct from the target object, but looks nearly the same.)

The following modifications to the algorithm G are needed to deal with imperfect edge finding. First of all, G must maintain estimates of the reliability R and the plausibility P of the current match in the course of its search. When R exceeds a predetermined threshold, the match should be accepted, and when P falls below another predetermined threshold, backtracking should occur. Second, G needs to perform a case analysis according to whether each expected edge is detected by the edge finder, in addition to the case analysis which it already performs concerning whether the edge is in view. This case analysis will also constitute a choice point for the purpose of back-tracking. Thus, G will proceed as follows. For each new edge e which it selects for matching, it will (1) assume that e is in view, (2) assume that e is detected, (3) look for e, (4) continue the match. If the match failed, then it will backtrack to (2) and assume that e, though in view, was not detected, and will proceed to the matching of other edges. Finally, if this last match fails, it will backtrack to (1) in the manner described earlier.

The algorithm as it now stands is no more than an elaboration on the simplest of matching algorithms: sequential matching with backtracking. Nonetheless, if we judge the efficiency of a matching method by the number of matching steps which it goes through in searching for a correct match, the algorithm does not come out badly. The principal reason for this is that only a few edges on an object of known shape need to be identified in order to determine the position and orientation of the object. In fact, identification of the image projections of three non-colinear points on the object is sufficient to narrow the set of possible positions and orientations of the object to at most two distinct possibilities. (This last statement holds exactly for orthographic projection, and applies to perspective projection as well unless the camera is very close to the object, or the precision of measurement is very high, in which case one of the two possibilities may be eliminated). A fourth non-coplanar point suffices to remove the remaining ambiguity. The identification of three pairwise non-parallel lines (without specified end points) will accomplish the same task. Thus, a match does not need to proceed very far before the locus of possible positions of the camera will have been narrowed to only a few grid points by back-projection. Thereafter, the matching of additional edges serves to check the correctness of the match, but not to further refine the estimate of camera position. The positions of these additional edges will be predicted accurately, and as a consequence the probability that many such edges will be found in the case of an incorrect match will be exceedingly low. Thus, bad matches are likely to fail very early. Conversely, the reliability of a good match will rise quickly as more expected edges are found, so that the cost of achieving a very reliable match is low. So, the effectiveness of the current algorithm relies on the fact that it requires exact quantitative matching of object edges to image edges at all stages during a match.

4. Specialization and instantiation of the one stage algorithm

The result of unwinding the main loop of the schematic algorithm G described in the last section may be diagramed informally in the following way.

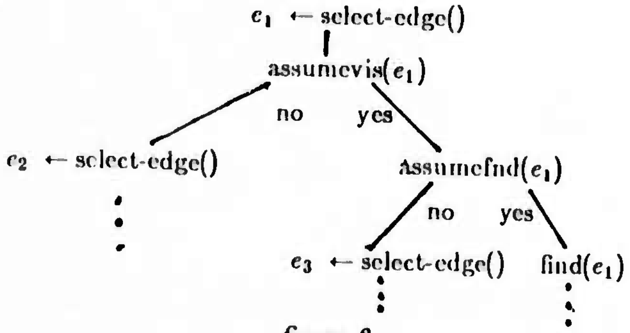


figure 2

Here, "find(e)" represents the predict-observe-back-project operation by which new edges are matched, "e ⇐select-edge()" represents the selection of an unmatched edge to look for, "assumevis(e)" represents the case analysis according to visibility of edges, and "assumefnd(e)" represents the case analysis according to whether the edge has been detected. Our job now is to fill in the details in this unwound schematic algorithm, making use as appropriate of the fact that the object description is available in advance. In the following discussion, we will employ the "compile-time/run-time" terminology familiar from compiler design. Operations which are carried out in the course of constructing specialized variants of G will be refered to as "compile-time" operations, while operations carried out by those specialized variants in the analysis of images will be refered to an "run-time" operations.

The compile-time process by which the above schematic search tree is filled in may be thought of as moving from the root of the tree down, fully instantiating nodes as it goes. Imagine for the moment that the first k levels of the tree have been filled in, and that the task at hand is to fill in a particular node at next level. As will be seen in a moment, selection of the object edge to be matched at each point in the tree is done at compile-time. That is, the select-edge() operation is "executed" at compile-time, so that each node in the instantiated search tree will refer to a particular object edge to be matched. The tree developed to level k will look something like this:

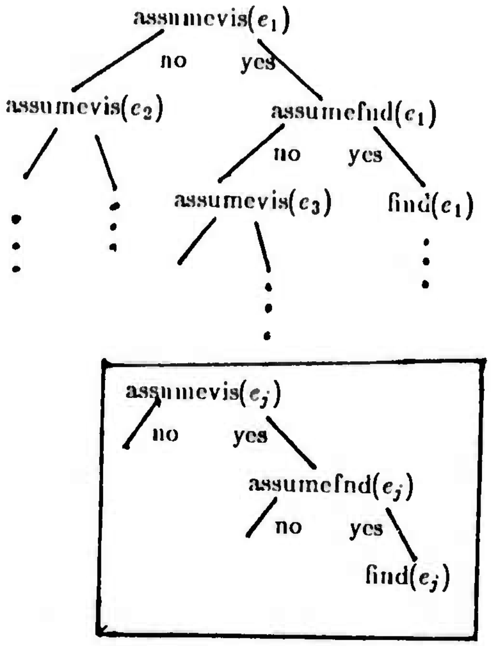


figure 3

The box indicates nodes to be filled in at the current stage. We will deal with all the nodes involved in matching a particular object edge at once, rather than following a strict level by level order. In what follows we will specify how each of the operations in the boxed nodes are instantiated.

(1) The find operation. This involves prediction of the position of e_j, a check to see if any image edges lie in the predicted position, and back projection. Recall that prediction is carried out by computing bounds on the location of e_j relative to an already matched edge e_0, and then using the known position of e_0 to get bounds on the position of e_j in the image. The position of one edge e_j relative to another e_0 may be specified in various ways. The only requirement here is that the relative position be given in a way that is invariant under translations and rotations. In any case, a vector of four numbers $[a_1, a_2, a_3, a_4]$ suffices. For example, a_1,a_2 might give the coordinates of the center of e_j relative to the image coordinate system with origin at the center of e_0 and x-axis directed along e_0, a_3 the length of e_j, and a_4 the orientation of e_j relative to e_0. Let $relpos(e_j, e_0, p)$ denote the position of e_j relative to e_0 from camera position p in whatever representation is chosen. More generally, let $relpos(e_j, e_0, K)$ denote bounds on the components of $relpos(e_j, e_0, p)$ as p ranges over locus K. Let L be the currently hypothesized locus at the time that the find operation is executed. We

want to compute $relpos(e_j, e_0, L)$. The following very simple scheme is adequate. Namely, at compile time, $relpos(e_j, e_0, g)$ is computed for each grid element g, and stored in a table. Then, $relpos(e_j, e_0, L)$ is computed at run-time simply by taking the union of the bounds $relpos(e_j, e_0, g)$ for $g \epsilon L$. These unions are taken component-wise, so that the end result of the process is a set of numerical upper and lower bounds on each of the components of the relative position. Note that this manner of computing bounds loses some information since constraints relating different components of the relative position are not expressed by simple bounds on the components. As a result, edges with positions predicted in this way may not be consistent with any camera position in the locus. But this is comparatively unlikely, and, in any case, bad edges of this kind will be thrown out in the back-projection stage.

The above method is a very crude, but it can be made quite fast. For example, less than 50 machine instructions per grid element are required to carry out the prediction operation in an aggressively coded implementation on the PDP-10, VAX, or Motorola 68000. The number of grid elements which need to be considered in a given prediction step of course depends on the details of the match in progress. In most matches, the size of position loci to be considered decreases rapidly as the match proceeds. For the experiments described later, the average prediction step involved less than 10 grid elements.

Efficient implementation of the observation stage is a standard exercise in computational geometry. The problem is to design a data structure for storing image edges such that the set of edges satisfying a given prediction can be retrieved rapidly. Any of a variety of methods involving binary search on the parameters of the prediction will do.

The tables of relative positions constructed at compile time for the prediction step can be used for back-projection as well. In order to determine which grid elements of the current locus are consistent with a given set of measurements, it is only necessary to compare the measured values to the bounds $relpos(e_j, e_0, g)$ which have been pre-computed for each grid element g.

(2) Selection of the next edge to look for.

In the course of a match, the currently hypothesized locus of camera positions is refined in two ways: by back-projection, and by making assumptions about the visibility or invisibility of particular edges. The data necessary for the latter kind of refinement is available at compile time. As a result, each point in the instantiated search tree has an associated compile-time locus of possible camera positions - namely, the set of camera positions which are consistent with the visibility assumptions made on the path leading from the root to the current node.

Our task is to select at compile-time an appropriate object edge to look for next at the current stage of the match. There a.e three considerations which are relevant to this selection. First, the likelihood that the selected edge is visible should be as high as possible. We don't wish to select an edge which will be visible from only a small fraction of the current locus or which, even if visible, the edge finder is unlikely to detect, since the computation time spent looking for it will then be unlikely to pay off. Second, the prediction of the position of the edge should be as specific as possible, since this will lead to a lower likelihood of false matches for the edge. Third, it is desirable that measurements on the image position of the observed edge should provide as much information as possible about the camera position. Each of these factors can be evaluated in a quantitative manner at compile time. Assuming a uniform probability distribution on the position of the camera (or more generally, assuming any particular prior distribution on camera positions) and statistics about the performance of the edge detector, the probability of the visibility of any edge over any locus can be computed. Similarly, the specificity of a prediction is naturally measured by the inverse of the probability that a randomly chosen edge will meet the prediction. This in turn can be computed in a straight-forward way from the bounds involved in the prediction. The numerical values of the bounds for the prediction can be computed for each possible camera position in the compile-time locus at compile time. By averaging these bounds (using the weighting of the prior distribution on camera positions), an expected specificity for a prediction of a given edge can be determined at compile time. Finally, in a similar manner, the information obtained from measuring the position of any given edge can be evaluated for each camera position at compile time.

By this method, the best edge to match next can be determined at compile time, assuming that a way of combining the factors listed above has been chosen. The question of exactly what weight should be given to each factor is a complex one. In qualitative terms, the order in which the considerations have been stated here reflects their order of importance for the efficiency of the matching process. Any weighting function which respects this order of importance is likely to be acceptable. We regard the detailed analysis of this question as an open research topic.

Note that the determination of the best edge to match next at any stage is computationally expensive, since it involves calculations concerning each edge at each camera position. But, as remarked above, large amounts of computation time at compile time are often justified for smaller gains at runtime.

(3) The visibility case analysis. The method by which visibility case analyses are performed has already been fully specified, so no compile time instantiation of the node is needed. However, certain visibility case analyses can be dispensed with entirely based on information available at compile time. It will often happen that the particular edge e_j whose visibility is in question is in fact visible throughout the compile-time locus, so that no case analysis according to its visibility need be performed. That is, it will often happen that along the

path taken from the root of the search tree to the current node, visibility assumptions have been made which together guarantee the visibility of the current edge. In this case, the case analysis node is simply left out.

Note that this optimization is a very important one in that it greatly reduces the size of the search tree (though usually has only a minor effect on its depth). If n edges are involved in a match, then in principle, there are 2^n distinct combinations of visibility and invisibility for the edges to be considered. Of course, only a small fraction of these cases can actually occur, since since the visibility of edges are not determined independently. For example, there are 26 distinct visibility/invisibility combinations for the edges of a cube (one for each face, edge, and vertex which the viewer might be "facing"), rather than $2^{12} = 4096$.

(4) The detection case analysis. We treat detection case analyses in a similar manner to visibility case analyses: we drop a detection case analysis if the assumption that the current edge is not detected causes the plausibility of the current match to drop below threshold. The information about the current match which is needed to compute its plausibility - namely, the information as to which edges have been detected and matched and which have not - is available at compile-time. It can be read off by following the path from the root of the search tree to the current node.

We can sum up the speed gains achieved by specialization in a very rough way by noting that the time required for an average matching step in the specialized program will be on the order of a few miliseconds on a 1 MIP machine. As a consequence, several hundred matching steps can be executed per second in the search for a match. This speed is much greater than that obtained by existing methods, and is adequate for a variety of practical applications.

5. Experiments

The scheme for generating special purpose vision programs described above has been implemented in MacLisp running on the PDP-10 at the Stanford Artificial Intelligence Laboratory. A few refinements not described earlier are included in the implementation. For example, in this account, we have not considered the effect of measurement error, nor have we described any method for matching partially visible (or partially detected) segments. The implementation includes machinery for dealing with both of these matters. Also, until now we have required that the distance to the object be known exactly in advance. This requirement is weakened in the implementation; it is generally sufficient if the distance is known to within a factor of 2.

On the other hand, the implementation does not yet fully automate the selection of the order in which edges are treated; in the experiments we chose the order by hand. Nor does it come close to realizing the potential for speed of the underlying algorithm. For example, efficient data structures and accessing methods for the set of image edges have not been implemented. The speed figures given earlier are estimates of what could be obtained in an aggressive implementation, not measurements of current performance.

So far, tests involving three different objects have been run. The objects treated were a connecting rod casting, a universal joint casting, and a key-cap (key-caps are the plastic keys which make up typewriter and terminal keyboards). In each test, a special purpose program was generated automatically from a description of the object; this program was then applied to images of the object digitized from a television camera. In the case of the connecting rod and universal joint, the pictures contained only one instance of the object against a relatively uncluttered background. These images were successfully analyzed with relatively little effort by the vision programs; correct matches were obtained in each case after fewer than 50 matching steps.

The special purpose program for recognizing key-caps was subjected to a more arduous test. We digitized an image of a jumbled pile of key-caps (see figure 4). The edges found in this image by David Marimont's edge finder [Marimont 1982] are displayed in figure 5. The task of the key-cap recognition program was to find instances of key-caps which were - roughly speaking - within 45 degrees of right-side-up. More precisely, the locus of allowable orientations of the camera relative to the key-cap was the locus making up the top face of the cube in the scheme for representing loci described earlier. Key-caps of a variety of shapes appear in the image; only key-caps with square upper faces were sought by the program. This is a severe test for the matching method for several reasons: (1) Objects of the desired kind must be recognized in a complex background - a background in which many objects similar to the target object appear. (2) The target object has only a limited number of features on which the match can be based. (3) Resolution is quite low. Although the entire image has a resolution of 240 by 240 pixels, each object to be recognized occupies only a 40 by 40 region. Also, lighting and contrast were not particularly good.

The program was run in a mode in which not just one, but every match meeting the reliability criteria was returned. Further, the reliability threshold was set at a very low level, so that every plausible match was found. The matches found were then ranked by reliability. The top three matches in this ranking were in fact correct. They are displayed in figure 6. Most of the remaining matches - which had been assigned lower reliability - were incorrect. The total number of matching steps required in this experiment was 960; so, in the hypothetical "aggressive implementation" mentioned earlier, the whole process would take a couple of seconds.

This experiment indicates that the matching algorithm can find matches under difficult circumstances.

In this case, due to the small number of matching features, it is not possible to achieve very high reliability of matches. Typical industrial objects, such as the castings mentioned above, have many more features on which a match can be based, and hence allow very good reliability of detected matches.

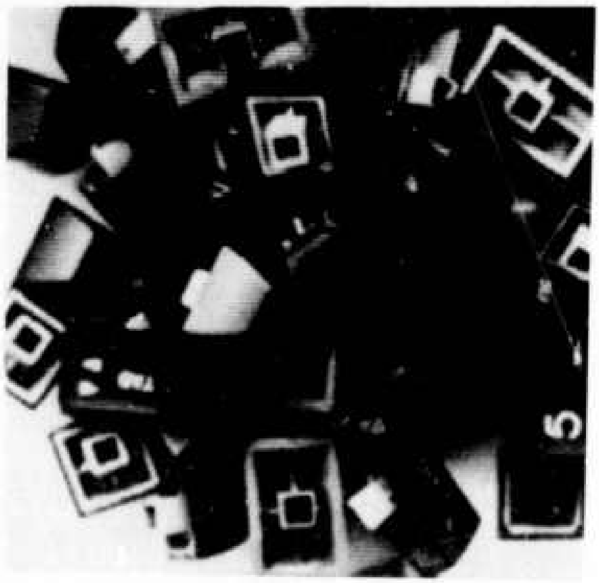


figure 4

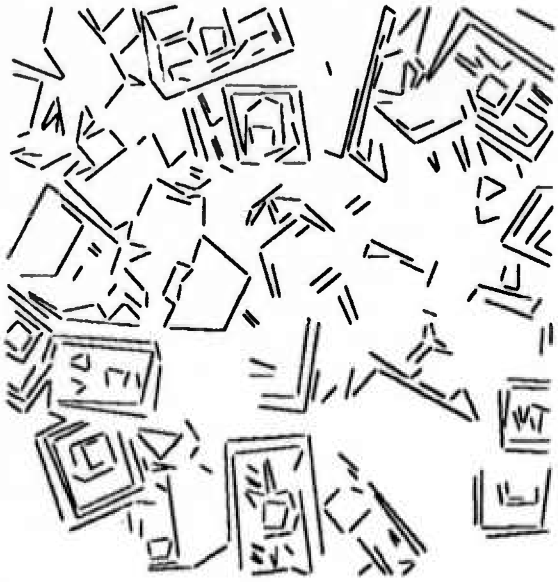


figure 5

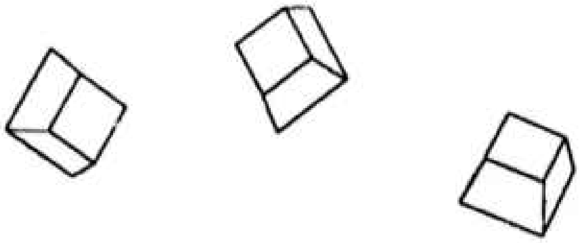


figure 6

6. Extensions

We have described in some detail the construction of specialized variants of a comparatively simple matching algorithm. It should be evident that the same general scheme can be applied to more complex matching algorithms which handle a wider class of problems, or which exploit additional structure in the matching situation in order to enhance performance.

An elaboration of the current algorithm which is particularly useful and to which our scheme for specialization extends easily is as follows. If the target object has symmetries, or if more generally there are recurring patterns of edges on the object, then the matching process may proceed by first seeking an instance of the recurring pattern, and then performing a case analysis according to which of several instances of the pattern has been encountered. This modification will speed up the matching process in the cases to which it is relevant, since a good match is likely to be found sooner, and since extensive bad matches to the "wrong" instance of the pattern will be avoided. The same technique can be used for matching of multiple objects which share common patterns of features. Again, the common pattern is matched first, and then a case analysis as to which object the pattern arose from is performed. The technique may be applied recursively to very large sets of target objects which have been classified in a hierarchical manner according to a taxonomy of common features. The taxonomy is exploited by a matching method which performs a kind of binary search down the hierarchy until a complete match is found.

More generally, the matching algorithm which we have considered is an instance of an extremely common kind perception algorithm. Such algorithms are built up from interleaved observation steps, in which some detectable quality of the world is predicted, observed, and used to refine the current world-model, and case analysis steps, in which assumptions are made about the world - assumptions which are not justified by any data or argument, but which are necessary to decide on what observation to make next, and which can be withdrawn later if necessary. This kind of algorithm appears - usually in elaborated form - in many areas of computing. Any such algorithm can be specialized according to the general plan which we used here. To perform the specialization, we proceed by first unwinding the case analysis steps into a full tree of possibilities. Then, we use the context of assumptions available at any point in this tree to optimize the work performed at that point.

7. Related Work

The particular vision problem which we have chosen to attack is 3D model-based vision in its strict form: the exact shape of the object to be recognized is assumed to be known in advance, and no restriction is placed on the 3D orientation of the object relative to the camera. Comparatively little work has been devoted to this form of the vision problem. There appear to be two traditions of work in model-based vision, one of which might be referred to as 2D vision from exact models, and the other as 3D vision from inexact models. The first tradition includes work focused directly on industrial problems such as that of Perkins [Perkins 1982], where the exact shape of the object is specified in advance, and where the orientation of the object is restricted in such a way as to reduce the problem to a "nearly" 2D form. The second tradition treats problems in which orientation is (comparatively) unrestricted, but where the previously available information about the model is less complete. Examples of this kind of work include [Garvey 1976] and [Shirai 1978]; here the matching processes used tend to employ qualitative rather than quantitative restrictions on matches. Acronym[Brooks 1981] is an exception to the above, in that it does exploit quantitative restrictions on the parameters involved in a match. Acronym uses a considerably more ornate matching scheme than ours. Also it uses a method for generating numerical constraints which is much more general and consequently much slower than ours - by a factor of at least 100. (Acronym does not "compile" the object model into a fast program as we do). Still, there are strong similarities in approach between our work and the work on Acronym.

Thus our work is less ambitious than some previous work in 3D model-based vision, in that we restrict ourselves to exactly specified models, and in that we are investigating comparatively simple algorithms. Nonetheless, it seems to us that the problems and processes involved in simple sequential matching of exactly specified models are not yet well understood, and that, as a research strategy, it makes some sense to concentrate on this limited domain before attacking matching problems of a more general kind.

The general strategy by which we have obtained an efficient implementation of matching - namely special purpose automatic programming - has been followed in technically different form by [Bolles 1982]. Bolles has attacked the problems of matching 2D models to images, and more recently, 3D models to range images, by what he calls the local feature focus method. The method involves selecting a class of "focus" features of similar shape on the object from which the match is to begin. Then maximal sets of mutually consistent interpretations for features near a given candidate match to a focus feature are sought. Such a "maximal clique" of consistent interpretations forms the seed for a more complete match, which is done sequentially. (See [Bolles 1982] for a description of the method). This method is compiled into a very fast matching program by the same kinds of methods we have used. Local features to be matched are selected in advance, and tables of relative positions are compiled. Experiments have shown that the maximal clique method is robust and fast for 2D matching.

The maximal clique method does not extend easily to the problem of 3D matching from intensity (rather than range) images. There are two reasons for this. First, the maximal clique method depends on the transitivity of the consistent-interpretation relation: if interpretation A for point a is consistent with interpretation B for point b, and if interpretation B for point b is consistent with interpretation C for point c, then interpretation A for point a is consistent with interpretation C for point c. This transitivity holds for 2D, and for 3D points from range data, but not for 2D projections from a 3D model with arbitrary orientation. (Still, the clique method might be used for more complicated structures for which this transitivity does hold). The second and more decisive reason for the difficulty of extending the maximal clique method to 3D is this. The method depends on the possibility of selecting a reasonably small set of image features which are candidates for matching features near the already matched focus feature, and which together uniquely identify the match. In 2D or 3D from range data, the identification of such a small set of features is aided by the fact that the positions and orientations of the local features relative to the focus feature are known in advance. In 3D intensity images this kind of advance information is not available, or is much weaker, so that the set of nearby features in the image which require consideration may be quite large. Equally importantly, the number of possible interpretations for each such feature will be large as well (the size of the graph of possible interpretations in which cliques must be found is of order $k \times n$, where k is the number of local features considered, and n is the average number of interpretations of each feature). In the keycap example, the interpretation graph would be so large that clique finding would be impractical. Sequential matching is less vulnerable to this k'nd of problem because at each stage in the match all of the information derived from the match so far is used to restrict the number of candidates for match at the next stage.

Bolles' work is very closely related to ours in general aim and stategy; his results support the idea that the feature matching approach to vision is feasible and robust.

Acknowledgements

We would like to thank David Marimont and David Lowe for many useful discussions. This research was supported in part by the National Science Foundation under Grant MCS81-04873 and in part by the Advanced Research Projects Agency of the Department of Defense under Contract MDA903-80-C-0102

References

[Bolles 1981]
Bolles, R.C., and Cain, R.A., *Recognizing and locating partially visible objects: the local-feature-focus method*, International Journal of Robotics Research, Vol 1,No. 3,Fall 1982

[Brooks 1981]
Brooks, R.A., *Symbolic reasoning among 3-D models and 2-D images*, Artificial Intelligence Journal, August, 1981.

[Garvey 1976]
Garvey, T.D., *Perceptual strategies for purposive vision*, Technical Note 117, AI Center, SRI International, 1976

[Goad 1982]
Goad, C. A., *Automatic construction of special purpose programs for hidden surface elimination*, Computer Graphics, vol. 16 No. 3, July 1982, pp. 167-178

[Marimont 1982]
Marimont, D.H., *Segmentation in Acronym*, Proc. Image Understanding Workshop, September 1982

[Perkins 1978]
Perkins, W.A., *A model-based vision system for industrial parts*, IEEE Trans. Comput. C-27:126-143.

[Shirai 1978]
Shirai, Y., *Recognition of man-made objects using edge cues* in **Computer Vision Systems**, A. Hanson, E. Riseman, eds, Academic Press, New York, 1978.

MAPS: The Organization of a Spatial Database System Using Imagery, Terrain, and Map Data

David M. McKeown, Jr.
Carnegie-Mellon University
Pittsburgh, PA 15213
April 1983

three dimensional

Abstract

This paper presents the system description and organization of MAPS, the Map Assisted Photo interpretation System. MAPS is a large integrated database system containing high resolution aerial photographs, digitized maps and other cartographic products, combined with detailed 3D descriptions of man-made and natural features in the Washington D. C. area. A classification of image database systems into three models is also presented. These models are the Image Database (ID) model, the Map Picture Database (MPD) model and the Image/Map Database (IMD) model.*

1. Introduction

This paper presents the system description and organization of MAPS, the Map Assisted Photo interpretation System. MAPS is a large integrated database system containing high resolution aerial photographs, digitized maps and other cartographic products, combined with detailed 3D descriptions of man-made and natural features in the Washington D. C. area.

This paper discusses three major topics. First, a classification of different models of database systems for cartographic applications is presented together with a discussion of their inherent strengths and limitations. These models are the Image Database (ID) model, the Map Picture Database (MPD) model and the Image/Map Database (IMD) model. Second, the author argues we argue for the utility of the Image/Map Database model, discuss tasks and present a general description of the model. This model describes components, facilities and techniques that should be present in such a system, and a range of tasks that can be supported by the model. Finally, he we describe the MAPS system in terms of the our (IMD) model, and discuss three applications which utilize and integrate image, terrain, and map data in a powerful manner. He We also discuss what has been we have learned during the implementation of the MAPS system, some ideas on the proper interfaces between components, where modularity should be achieved, and point to future work.

2. Background

Our early motivation for investigating image databases was as a component of a complete image understanding system. We had only a vague idea of what capabilities it should have, but we thought that it should represent "idealized segmentations" of an image, where the labeling of the segments was in fact the "scene interpretation". It should relate, or compare machine generated segmentations to this model, and provide the user with a qualitative and quantitative performance measure of the machine segmentation. We attempted this with the MIDAS system[1, 2] using the segmentation results for a set of Pittsburgh city scenes generated by the ARGOS[3, 4] system. The results of the performance analysis of the scene segmentation were less than encouraging. While we could give quantitative analysis of the segmentation and labeling by the ARGOS system, the qualitative results were couched in the original (subjective) hand segmentations. It was difficult to qualitatively distinguish between alternative machine segmentations, since the relative importance (or cost function) of missing or mislabeled regions or broken boundaries for different regions was not represented in the segmentation. How to perform such an evaluation is still an open research problem. Also, although we had a database of 18 high resolution color images of Pittsburgh, we had no general mechanism to relate one to another, except through analysis of the hand segmentations and the names given to buildings, roads, rivers, and other features in the scene. However, in the process of implementing and using MIDAS we did learn a great deal about image database organization and symbolic representation of scene descriptions.

We decided to look at map-guided image interpretation and began to assemble an aerial photograph database of the Washington, D. C. area. Using this imagery, we felt, we could quickly generate a map database that would allow us to explore image analysis of complex aerial photographs using a simple map database that constrained where to look, and what to look for. This idea of map-guided segmentation was not new. The HAWKEYE system[5] and succeeding "road expert"[6, 7] were based on similar ideas, and use of world knowledge had been a well accepted paradigm in image interpretation. However, we wanted to focus on more general capabilities, to represent large scale spatial organizations normally encountered in complex urban scenes. The generation of the map database turned out to be a much harder

*This research was sponsored by the Defense Advanced Research Projects Agency (DOD), ARPA Order No 3597, monitored by the Air Force Avionics Laboratory Under Contract F33615-81-K-1539. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

problem than we initially estimated, and it quickly became the focus of our research. In retrospect, I believe, it was exactly the right problem to work on, and although there is still much to do in the area of image/map databases, we now have the right tools and understanding to begin to tackle the original problem. This work has direct application in three areas:

- photo-interpretation: representation of world knowledge for image understanding.
- situation assessment: a spatial expert for decision support systems.
- cartography: toward digital map generation and use.

3. Classification of Databases

There has been, over the last ten years, a perceived need for organizing and structuring image and map data for cartographic applications. It has been difficult to compare various capabilities and limitations of systems because there were few common denominators by which systems could be compared. Systems reported in the literature could loosely be categorized either as research vehicles, or production-oriented systems for particular well defined subtasks of the general cartographic problem[8, 9, 10]. Research vehicles generally had a high degree of organizational complexity tested on very small scale databases. Systems used in production environments tended toward simple models running very large scale databases. Further, while the tasks being performed involved the analysis of aerial or satellite data, it is often unclear whether the image data was an integral part of the resulting database, or simply used for data acquisition. One example is the development of digital filing systems that store facts about a large number of images without storing the actual image data. The best example of such a system is the EROS Data Center database maintained by the U.S. Dept. of the Interior. This database has approximately $2x10^6$ frames of Landsat imagery and $5x10^6$ frames of aircraft (aerial mapping) photography. Users may specify an area of interest by geodetic point or rectangular area and sub-select those frames based on time of year, cloud cover, type of sensor[**] and a a scene quality rating. However, the actual frames of data are stored on high density magnetic tape. Similar situations exist in map producing organizations such as the United States Geological Survey (USGS) and the Defense Mapping Acency DMA.

One notable exception is described in Kondo et.al.[11] where an image database using Landsat imagery was integrated with map descriptions for geographic, natural, and cultural features. Features can be displayed superimposed on the image data, and imagery could be indexed by geodetic location or by feature name. There are limitations such as: the image-to-map correspondence was based on a fixed decomposition of landsat data into a latitude/longitude grid at a map scale of 1:50000; the spatial relationships between features were entered manually; and the overall complexity of the image and map database was small. Nevertheless, this represents an ambitious new direction for the development of land-use systems using Landsat imagery.

In this discussion of database systems for cartographic and situation assessment applications, we are assuming that the following minimal capabilities hold: (1) on-line display of digital imagery and map data, and (2) ability to query interactively about attributes of the imagery and map. The following is our classification of the capabilities of three models which we can use to compare various existing systems or approaches. These models are the Image Database (ID) model, the Map Picture Database (MPD) model and the Image/Map Database (IMD) model.

3.1. Image Databases

The Image Database model (ID) is the simplest and most common database model. It is organized to relate attributes about the sensed image such as sensor-type, acquisition, cloud cover, or geodetic coverage[***]. These databases generally do not represent the content of the scene, but rather attributes of the scene. When the semantics of the scene are present, the location of cartographic features are represented in the image (pixel) coordinate system. This poses obvious limitations to the application of relevant knowledge from other images or from external sources, since there is no general mechanism to relate map feature position between images that overlap in coverage or to an external map. Although the features represented may appear to be map-oriented, is is difficult to compute general geometric properties using the image raster as the coordinate system.

Although relational database techniques have been applied to the ID model, we feel these techniques are not appropriate to spatial database organizations for several reasons. First, using the basic *<attribute, value>* tuple to represent vector lists of map coordinate data requires that all of the primary key attributes be duplicated in each relation, since there is no mechanism for allowing multiple valued (sets, lists, order pairs) as a primitive attribute in a relation. Further, the relational database operations such as union, intersection, join, project, are not good primitives for implementation of inherently geometric operations such as containment, adjacency, intersection and closest point. Operations such as feature intersection are reduced to searching for line segments which share the same pixel position. Finally, in any large system, a logical partitioning of the database must be performed in order to avoid extensive and often unnecessary search when performing spatial operations. Partitioning is difficult to achieve in relational systems since the relational model restricts itself to homogeneous (only one record type) sequential sets. Previous work advocating such organizations did not address the issues of system scale, and focused more on issues of query languages using relational models for geographic databases than the actual construction of complex systems[12, 13, 14]. When measured by the number of images, image-based features, and by the complexity of the relationships represented, these systems were quite simplistic.

[**] for aerial mapping photography

[***] using frame annotation such as the center point and corner points not using general image-to-map correspondence

3.2. Map Picture Databases

The Map Picture Database model (MPD) describes databases that are generated by digitizing cartographic products, such as pre-existing maps and charts. These databases are attractive in environments where paper maps have played a large role in planning and analysis. There are, however, some major limitations to spatial systems based on digitized cartographic products. First, in the original map production, spatial ambiguity has been rectified by the cartographer in a manner that is not often reversable. The cartographic process involves simplication (generalization), classification (abstraction), and symbolization of real-world ambiguity. Constraints imposed by the scale of the map often determine which world features can be depicted despite the desirability of portraying a complete spatial representation. Therefore, map icon and symbology placement may not be as accurate as the original source material. Since the deduction of the actual spatial arrangement of objects from an iconic representation is an open problem, MPD's represent chaos masquerading as rationalized order. The key issue is that MPD's are pictures of a map (however detailed) rather than the underlying map structure and spatial organization. Although the graphics display of MPD appears to convey a great deal of semantic information, that impression is a result of the human observer, not a reflection of an underlying map representation.

When a map is digitized into a map picture, another subtle simplification occurs. The digitization process results in a map image on a rectangular grid whose size is generally limited either by custom or as an artifact of the digitization process. Common limitations are scanner resolution, maximum size of image raster, and the physical size of source map. One popular representation is to subdivide regions of the map picture into a regular decomposition such as quad-tree[15, 16], or k-d tree[17]. The implementation of this representation is greatly simplified in MPD models since one no longer has to contend with positional ambiguity of map features because of the cartographic process outlined above, and the discrete nature of the digitization process.

One common use for the MPD model is in geographic information systems for land use and urban planning. In these systems, aggregrate values such as population of an area and crop yield of an area are computed. The scale of the original map becomes the limiting factor for accuracy in information computation. However, the grain of computation is usually large enough that these inaccuracies are not a practical problem. Incremental update of the database due to new residential and industrial areas and the concomitant loss of rural areas is a difficult problem since database update requires careful map editing tools not usually associated with these MPD systems.

A recent trend has been to take existing MPD databases and add a map feature database component, usually relational to describe attributes of various features. We believe that augmenting traditional MPD databases with semantic information has merit in those enviroments where analysis is being performed by humans, since information synthesis is not a requirement of the database system. However, once such a system is in place, there is a tendency to attempt to automate analysis functions requiring spatial interpretation, and the generation method of the MPD model has several drawbacks for use in photo-interpretation, situation assessment, and cartography. The chief problems are the method of generation as outlined above, the lack of semantic information about map features, and the requirement that a map exist at the appropriate level of detail for the area under consideration. The IMD model discussed in the following section addresses these issues.

3.3. Image/Map Databases

The Image/Map Database model (IMD) relates map features to image database through camera models. It therefore has the capability to describe relationships between features acquired from different images through the map database. This capability is in contrast to the image database model where the feature descriptions can only be related if the descriptions come from the same image.

Since the map database is built directly from aerial imagery in the IMD model, the resolution / accuracy issue is a function of the ground resolution of the imagery, the intrinsic position measurement error due to camera model, ground control, etc. rather than an artifact of the map depiction scale as in the MPD model. A greater variety of feature descriptions is possible since they are not restricted to those that can be portrayed in a cartographic product. Further, the complexity of a particular feature description is independent of any particular task requirement and can represent a rich set of attributes, semantic interpretations, and knowledge from diverse sources. This flexibility is a key element for map data representation as we look toward spatial database systems with applications in cartographic production, expert photo-interpretation, and situation assessment.

However, just as the cartographer must resolve ambiguity, so the spatial database must be able to represent inconsistency in a consistent manner. For example, errors in correspondence between images and the geodetic model cause the same point on the earth to be given a different geodetic position, ie. when viewed from different images the same geodetic point produces a different world position. If this point is on a common boundary between two features, say a political boundary, there should be ambiguity as to which region the point is in. By the same token, if two large residental areas are found to intersect because of positional uncertainty, and the result of the intersection is several small polygonal areas, the IMD model should be able to rectify this ambiguity. This rectification might take the form of a symbolic relationship that indicates that the residential area share a common boundary, while maintaining the ability to represent the original errorful signal data. Since the original data is maintained in the database, the symbolic relationships do not have to be static. For example, these relationships can be dependant on attributes similiar to those used by cartographers when they perform simplification and generalization. The link from the symbolic interpretation back to the original source data is not possible in MPD systems.

3.3.1. Spatial Knowledge

The IMD model gives us the tools to construct our map database from "first principles" and tie together partial spatial knowledge at different levels of detail. This is possible because individual map features may be specified directly from source imagery. This capability is precluded by the derivative nature of the MPD model. That is, it is difficult to assimilate new and possibly errorful knowledge because of the mismatch between the new errorful data and the cartographic rectification of ambigious data.

The representation of a multiple levels of detail paradigm is often invoked as a part of a coarse-fine or hierarchical matching strategy in image processing and interpretation. Given the scale and digitized ground resolution of an image, the IMD model can generate a map description that will suppress any features that would be too small to be recognized, with remaining descriptions at the appropriate level of detail. This technique is more than camera scaling and transformation, since the criterion for "too small" can be an attribute of the map feature itself. Consider the map feature description of a university campus. At some level of detail corresponding to pixel ground resolution distance (GRD), features such as playing fields, dormitories, instructional buildings and offices, access roads, and campus greenery are all individually distinguished. Using spectral properties of the features[••••] and spatial relationships between these features, we can determine those feature boundaries that are likely to be muddled, and those with sufficient detail to be recognized.

The multiple level of detail paradigm need not be applied in a homogeneous manner. For example, tasks such as decision aids for photo-intelligence may require high resolution detail to support analysis, but low resolution detail to establish overall context. A large scale spatial organization containing urban, residential, and rural areas will require flexibility to represent the high feature density and complexity in the urban area as well as significantly lower density in rural areas.

Flexible knowledge acquisition is necessary because in photo-interpretation, situation assessment, and cartography, world knowledge is inherently fragmented. Knowledge fragmentation in these domains arises from:

- **methods of knowledge acquisition**
  There are diverse sources of knowlege that are used to acquire map feature information. Some of the most common are direct measurement from imagery, old maps and charts, sketches, and collateral data.
- **task requirements**
  If the task requirement is to support radar scene simulation, then elevated roads are significant, and road networks in general are not significant. If the task is to support map generation at a particular scale (say 1:50000), the feature size density may determine whether it is directly portrayed, generalized, or omitted entirely. There are, of course, well defined rules that govern these decisions, but they are generally not consistent across a wide range of map scales.
- **specialization in feature extraction**
  There is a certain amount of specialization in cartographic and situation assessment activities. Analysts may specialize in a particular area of the world, be knowledgable in hydrology, geology, local construction customs, or political matters. In the production of large scale maps it is rare to find map generalists, although this may not be true for low level feature extraction activities. This specialization tends to fragment knowledge, and is often given as a justification for building database systems that provide access to a wide range of map knowledge and may have general capabilities for knowledge synthesis.

The IMD model methodology provides a mechanism for feature unification in a cohesive framework. It provides a framework to relate symbolic descriptions to their original data sources. It is not tied to a particular cartographic representation nor to limitations of cartographic production.

4. The Database Problem in Image Interpretation

The database problem has been addressed in a variety of ways in systems that perform image analysis and interpretation. However, it has rarely been pursued as a separate research problem. One explanation for this is that portions of general database represention are often embedded in the experimental image processing systems and become highly tuned to the application. This is sometimes a result of system performance issues, or ease of task-specific implementations, but often it is a result of not recognizing the database problem as a separate issue.

It is difficult to give a precise analysis of the use of map databases in image interpretation, since the detailed organizations of experimental systems are rarely available. However, there are several recent examples. Work at SRI used a map database of road intersections to construct a camera model in the HAWKEYE and subsequent "road expert" systems[5, 6, 7].

The ARGOS[1, 4] system used a digitized city plan map and elevations for buildings to build a 3D graphics model of downtown Pittsburgh. This model was directly compiled into a knowledge network representation which described size, shape and relative positions of buildings, roads, rivers, and bridges for an arbitrary view point. Although it was not tied to a geodetic grid, it was a general map model.

[••••] for example: roads preserve linear properties until the GRD approximately equals the width of the road

Recent work at Hughes[18] based on the ACRONYM system developed by Brooks and Binford[19] uses image registration to a geographic mod.el. The system uses pre-selected regions of interest and attempts to locate and identify pre-defined object instances within these areas.

ACRONYM is currently the best example of a model-based system that incorporates viewpoint-insensitive mechanisms in terms of its model description. Its recognition process is to map edge-based image properties to instances of object models. In the domain of aerial photo interpretation, results have been reported for the recognition of a small number of models (3) for wide-bodied jets in aerial photographs. It is not clear how map knowledge would be directly integrated into the ACRONYM framework, but one could speculate that it could be added by a method similar to the work at Hughes described above.

Matsuyama[20, 21] has demonstrated a system for segmentation and interpretation of color-infrared aerial photographs containing roads, rivers, forests, and residential and agricultural areas. It uses rules to make assignments based on region adjacency and multi-spectral properties. These rules make use of informal map knowledge but do not directly use a particular map to guide interpretation. It generates good descriptions of a variety of fairly complex aerial scenes getting a great deal of constraint from the multi-spectral data.

In his recent thesis, Selfridge[22] proposed using adaptive threshold selection for region extraction by histogramming and region growing using an image-based "appearance model". Although the work describes feature positions and shapes in terms of pixel descriptions, it is not difficult to imagine a more general map-based approach that would result in the automatic generation of constaints to his adaptive operators.

At CMU, Herman[23] has demonstrated the feasibility of incremental acquisition of 3D scene descriptions from stereo-pair aerial photographs in the MAPS database in the 3D Mosiac project. This system requires a known stereo camera model but uses no *a-priori* knowledge about the scene other than weak geometric assumptions about urban environments.

5. The Image/Map Database Model

In this section we discuss four classes of tasks that are common to photo interpretation, situation assessment, and cartography. We then list some criteria by which one can evaluate the strengths and limitations of database systems. These criteria are not exhaustive, rather they point to four areas that should be present in IMD implementations and system capabilities in each of the areas.

5.1. Tasks for Image/Map Database

In this section we give a classification of tasks that are common to applications in photo-interpretation, situation assessment, and digital cartography systems. The four tasks are *selection* of image, terrain, or map data based on attributes of the data, *spatial computation* of map feature relationships, *semantic computation* of map features, and *synthesis* of imagery, terrain and map data.

1. **Selection**

The selection task requires that the IMD system be able to select from a potentially large set of database entities based on attributes of image, terrain, and map database features. The selection task does not require image-to-map correspondence, and is the task normally performed by ID model systems. For example:

- select imagery with particular intrinsic characteristics: sensor, scale, date, cloud cover, processing history
- select map features based on symbolic description, partially specified description, similarities in image acquisition

2. **Spatial Computation**

Spatial computation is ubiquitous in cartographic, situation assessment and photo-interpretation tasks. An IMD system must provide tools to compute common spatial relationships such as containment, closest point, adjacency, and intersection. One issue is how to structure the environment in order to constrain search and thereby avoid unnecessary computation. Consider four views of the same problem:

- given a geodetic area, which images cover, or partially cover this area
- which roads can be found within the image
- which images contain this building
- given an image, find all images which overlap it

3. **Semantic Computation**

There are a number of tasks that require more than basic spatial computation, or where the appropriate spatial operation depends on the meaning of the map objects. Are there intrinsic high-level properties of map features that we can extract from basic spatial geometry that give a meaning to the feature? Semantic computation needs to be investigated as we develop more complex spatial databases. For example, what is the semantics of 'intersection' for the following pairs of map objects?

- intersection of two roads
- intersection of bridge and river description
- intersection of a building and a road

4. **Synthesis**

One goal of any database system should be to bring together diverse sources of knowledge into a common framework. Synthesis is the generation of new information using a new method of presentation, computation, or analysis. For example:

- cartographic superposition of map data on newly acquired image
- 3D display of terrain and cultural features from map database including man-made structures, political boundaries, neighborhoods, arbitrary collections of physically realized features
- to predict spatjal (location) and structural (appearance) constraints; where to look and what to look for based of task knowledge, previous experience, or expectations
- a spatial framework within which to embedd task-specific knowledge

5.2. Criteria for Image/Map Database

In this section we list some criteria that can be uses to evaluate database systems in four general areas. These areas are image-to-map correspondence, map feature representation spatial computation, and database synthesis.

1. **Image-to-Map Correspondence**

- can the it relate image-based features to a map coordinate system
- can these features be projected onto new imagery using the correspondence mechanism
- chat capabilities exist for incrementally updating feature descriptions based on updates to the camera model, or to intrinsic changes to the feature itself.

2. **Representation**

- what are the capabilities for feature representation; what complex spatial relationships can represent; how is inconsistency recognized and handled
- can the user describe features and associated attributes in a flexible manner; what is the variety of attributes.
- can the representation accommodate map-based information coming from a variety non-imagery sources
- what is the relationship between the representation of signal and symbolic data
- what synthesis tasks does the representation support

3. **Spatial Computation**

- does the system support dynamic spatial queries
- what spatial relationships does the system compute directly from the underlying data, which relationships are specified by the user, how do they interact, how does one maintain consistency
- what mechanisms are available to partition the search space when computing spatial relationships

4. **Database Synthesis**

- imagery, terrain and map data are components, each with an appropriate representation, operation semantics, and utility; in what ways does the database support synthesis of these components
- what concrete tasks requiring synthesis are performed

6. MAPS Overview

In the previous sections we have attempted to raise issues of Image/Map Database organization, tasks and capabilities. In this section we will discuss the MAPS system components capabilities. We will only briefly describe those aspects that have been reported on in other papers. Our latest work in the area of hierarchical organization, decomposition, and search is reported beginning in Section 6.6. New work in map feature semantics is discussed in Section 6.7. For a more detailed description of the image segmentation program (Section 6.1.2) and the image-to-map correspondence program (Section 6.3) see McKeown[24]. For a detailed description of the CONCEPTMAP database see McKeown[25]. Appendix I contains a nearly complete list of the programs associated with each system component.

6.1. BROWSE: Interactive Image/Map Display

BROWSE[26] is an interactive window-based image display system. It provides a common interface to all of the MAPS system components to display results of queries, graphical prompts for interactive image-to-map correspondence, superimpostion of map data on imagery, and other similar functions. While often viewed as an application issue, a flexible, functional user interface is critical for building more complex tools. BROWSE provides the user with a window-oriented interface, which greatly increases the effective spatial resolution of the frame-buffer, and provides multiple processing contexts which allow users to manipulate dynamically the size, level of detail, and visibility of imagery.

6.1.1. Window-based Display

We have applied and extended the bit-map window[27] paradigm to handle high resolution, multi-bit per pixel digitized images. However, due to nearly an order of magnitude difference in the amount of data needed to perform screen updates and due to processing limitations found in most frame buffer architectures, many of the solutions used for single bit per pixel displays[28] are not suitable for direct implementation. A detailed discussion of the design and organization of the window manager appears in McKeown & Denlinger[26].

Besides the display of imagery, we have found the window representation to be useful as a communication mechanism between MAPS components, to invoke image processing programs, and to retrieve and display the results of such processing. All MAPS components (see Appendix I) that display imagery, map data or graphics use the BROWSE window mechanism for display and communication. For example, the interactive image correspondence program. CORRES, uses the window mechanism to automatically display landmark image fragments and to create a high resolution window containing the approximate position of the landmark ground control point to cue the user. PICPAC contains a collection of image processing routines that can be invoked on BROWSE windows simply by specifying the window name. BROWSE routines use the window name to determine the image name, resolution, and rectangular image bounds. This information, along with parameters specific to the particular processing operation, are passed to the image processing routine. The results of the operation can be displayed in a new window.

6.1.2. Interactive Image Segmentation

SEGMENT is an interactive image segmentation program which uses the BROWSE window facility to provide an interface to our frame buffer. Users can extract image-based descriptions of map features, edit existing features, and assign symbolic names to the features. SEGMENT produces a standard format [SEG] file that is used throughout the MAPS database to represent image-based descriptions of point, line, and polygon geometric data. Database routines discussed in Section 6.5 are available to convert the [SEG] description to a map-based description [D3].

6.2. Image Database

The MAPS system currently contains approximately 100 digitized images, most of which are low altitude aerial mapping photographs. Typical ground resolution distances (GRD) are 120cm^2, 360cm^2, and 600cm^2 per pixel. The imagery is mainly comprised of three data sets taken in 1974, 1976 and 1982. In addition to aerial mapping photographs, we have several digitized maps including a USGS topographic map, and tour guide maps. Figure 1 gives the current status of the MAPS Washington D.C. image database. Although we have several Landsat, Skylab and high altitude aerial photographs taken over the Washington D.C. area, we have focused our work on those images that provide the greatest ground detail.

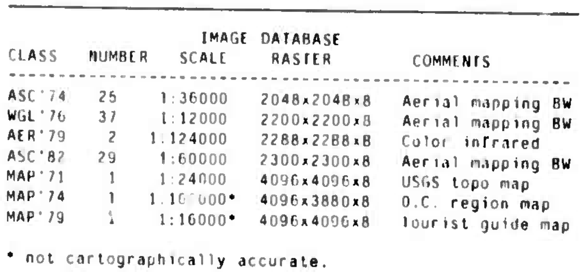


		IMAGE	DATABASE	
CLASS	NUMBER	SCALE	RASTER	COMMENTS
ASC'74	25	1:36000	2048x2048x8	Aerial mapping BW
WGL'76	37	1:12000	2200x2200x8	Aerial mapping BW
AER'79	2	1:124000	2288x2288x8	Color infrared
ASC'82	29	1:60000	2300x2300x8	Aerial mapping BW
MAP'71	1	1:24000	4096x4096x8	USGS topo map
MAP'74	1	1:100000*	4096x3880x8	D.C. region map
MAP'79	1	1:16000*	4096x4096x8	Tourist guide map

* not cartographically accurate.

Figure 1: MAPS: Image Database Component

6.2.1. Generic Image to File Mapping

The MAPS system uses a generic naming convention to refer to images in the database. The generic name is a unique identifier assigned to the image when it is integrated into the database. For example, DC38617, DC1420 are representative generic names that correspond to flight line annotation on the photographic film. All types of image access that require the filesystem name of the image, or require associated image database files, use the generic name mechanism to construct the appropriate physical file name. It is possible to change the logical and/or physical location of imagery by updating the generic name file or to add another image to the database. As we move to larger image/map systems this naming isolation allows us to construct a database that can be distributed over multiple

The decoupling of name with physical or logical location fits well with name server organizations usually employed with such distributed systems.

The following table lists the database files associated with each [illegible] image in the MAPS database. Each is accessible using the generic name.

- [GENERIC] **image-to-file system mapping**
  - contains the file system location of the database image
  - identifies which reduced resolution images are computed and available for hierarchical display
- [SDF] **scene description file**
  - contains image specific information: source, date, time of day, raster size, digitization, image scale, geodetic corner points, camera information
- [COF] **image-to-map coefficients file**
  - contains camera model coefficients, error model, polynomial orders solved, best correspondence (default polynomial order)
  - independent coefficients for <latitude>, <longitude>, <image row> <image column>
- [COR] **correspondence pairs file**
  - mapping of ground control points to image point specification
  - lists of landmark names and their geodetic position combined with image pixel position of landmark specified by user
- [HYP] **hypothesized landmark file**
  - lists of landmark names which are within the image geodetic coverage, but were not used to perform image-map correspondence

6.2.2. Image-Based Segmentations

MAPS maintains several types of image segmentations and map overlay descriptions associated with each image in the database. These segmentaions either are feature descriptions generated using the image as the base coordinate system, or the projection of map features onto the image using map-to-image correspondence, or segmentations from other images registered to the image. In the latter case, image-to-map correspondence is used to register the two images. Users can point to segmentation overlay features using the display interface in BROWSE and CONCEPTMAP, identify the segmentation feature name and retrieve its image and geodetic coordinates. For the [DEMSSEG] and [CONCEPTSEG] segmentation descriptions, the name of the segmentation feature is used to retrieve the associated DEAD (see Section 6.4) or

CONCEPTMAP description. The following table is a list of image segmentations associated with each image in the database. Segmentations that require map correspondence for their generation can be automatically recreated when image camera model is updated

- [HANDSEG] **hand (human) segmentation**
  - collection of all hand segmentations performed on this image
- [HCOMPSEG] **composite hand segmentation**
  - collection of all features in the [HANDSEG] database that are spatially contained in this image
- [MACHSEG] **machine segmentation**
  - collection of all machine segmentations performed using the image
- [MCOMPSEG] **composite machine segmentation**
  - collection of all features in the [MACHSEG] database that are spatially contained in the image
- [DLMSSEG] **DLMS map overlay**
  - all features from the DLMS digital feature analysis database that are spatially contained in the image
- [CONCEPTSEG] **CONCEPTMAP map overlay**
  - all features from the CONCEPTMAP database that are spatially contained in the image
- [COVERSEG] **image coverage overlay**
  - all images whose area of coverage is overlapped or wholly contained within the image

6.3. Image-to-Map Correspondence

The MAPS system uses an interactive image-to-map correspondence procedure to place new imagery into correspondence with the map database. It has three major components: a landmark database, a landmark creation and editing program, and an interactive correspondence program. The process of landmark selection, description, and interactive correspondence has been described in detail in McKeown[24].

6.3.1. Landmark Database

MAPS maintains a database of approximately 200 geodetic ground control points in the Washington D.C. area. Landmarks are acquired using USGS topographic maps, but in principle can be integrated from any source that provides accurate geodetic position *<latitude/longitute/elevation>*. Users can query the database to find landmarks by name, within a geodetic area, or the closest landmark to a geodetic point. Landmark features are also integrated into the CONCEPTMAP database and can be found using the *<role-derivation>* attribute (see Section 6.5.2) of a concept role schema.

6.3.2. LANDMARK

LANDMARK is an interactive tool used to generate new landmarks, their text descriptions, and associated image fragments. The following information is maintained by LANDMARK to support landmark database access.

- [LDM] **landmark name directory**
  - associates the list of landmark names with their geodetic position
  - sorted for spatial proximity
  - partial name matching also provided
- [LTX] **landmark text description**
  - contains a detailed text description of the location of the landmark and general factual properties of the landmark
  - stores the location and name of the associated image fragment file [LIMG], and replicates the geodetic position from ldm file
- [LIMG] **landmark image fragment**
  - contains a high-resolution image fragment which clearly shows the ground control point and scene context around the point

6.3.3. CORRES

CORRES is an interactive image-to-map correspondence program. It uses the BROWSE window interface, the LANDMARK database, and image database routines to interactively build an image-to-map correspondence. Once an initial guess of the corner points is performed and the [COR] and [COE] files have been created in the image database, CORRES automatically suggests new possible landmark points using the image database [HYP] files. The LANDMARK database [LIMG] files are used to display the ground control point when the user selects it from the list of hypothesized points.

6.4. DLMS: An External Database

The ability to rendezvous with externally generated map databases is a key capability in order to integrate information from a variety of sources. One example of the flexibility of the MAPS database is illustrated by our experiences with the Defense Mapping Agency's (DMA) Digital Landmass Simulation System (DLMS)[29].

DLMS is composed of a digital feature analysis database (DFAD) which describes man-made cultural features and a digital terrain elevation database (DTED) which is organized as a raster elevation grid. The specified resolution of the DFAD data is comparable to map scales of 1:250,000 to 1:100,000. The specified resolution of DTED data is within a meter vertical resolution over a 100^2 meter (3 arc sec) grid.

6.4.1. DFAD: Digital Feature Analysis Database

In order to integrate the DFAD database into MAPS, we reorganized the internal DFAD data structures to allow for random access using a feature header list. We converted the representation of geodetic coordinates from an offset format that was relative to an internal base coordinate, to an absolute coordinate system. Our DFAD database covers a two degree square area, from latitude N 38^0 to N 40^0 and longitude W 76^0 to W 78^0. It is composed of 64 "map sheets", each containing a 15'x15' map area. We assigned unique feature identifiers (names) to map features because feature numbers were not unique across map sheets. There are no feature names or semantics associated

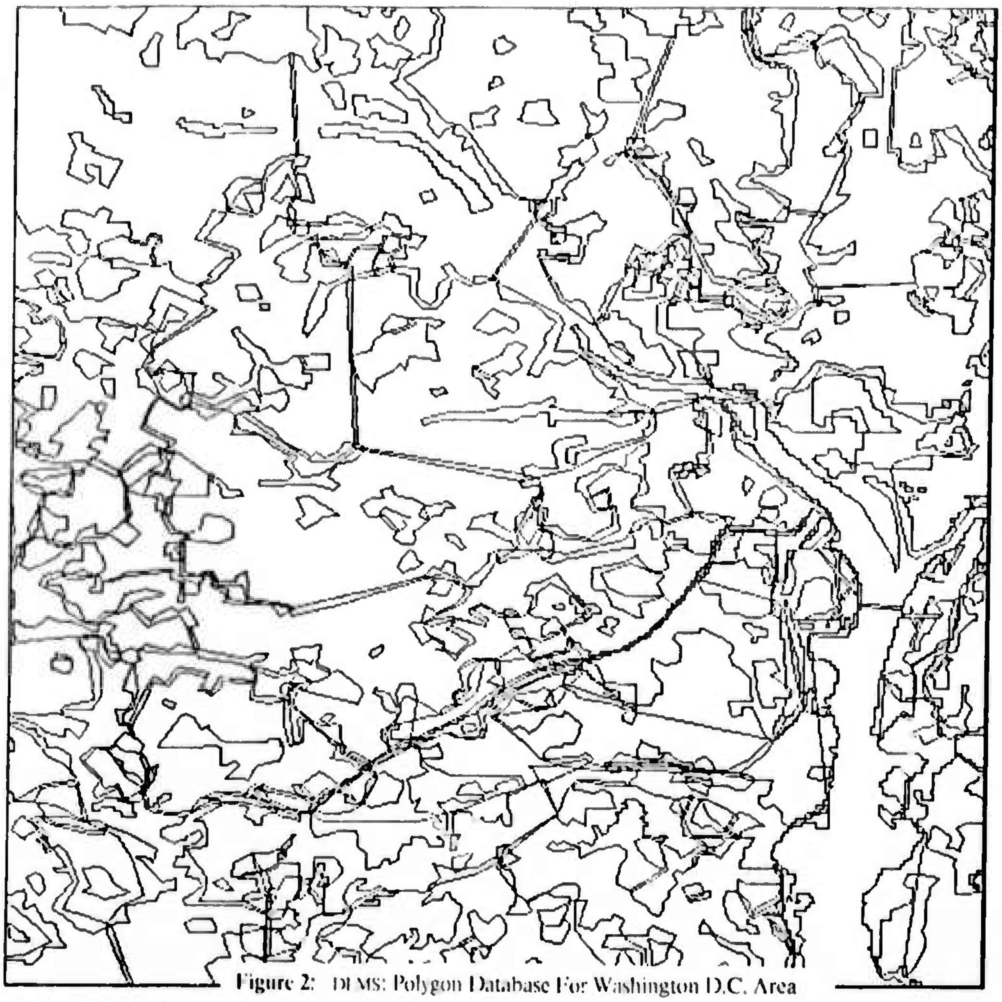


Figure 2: DLMS: Polygon Database For Washington D.C. Area

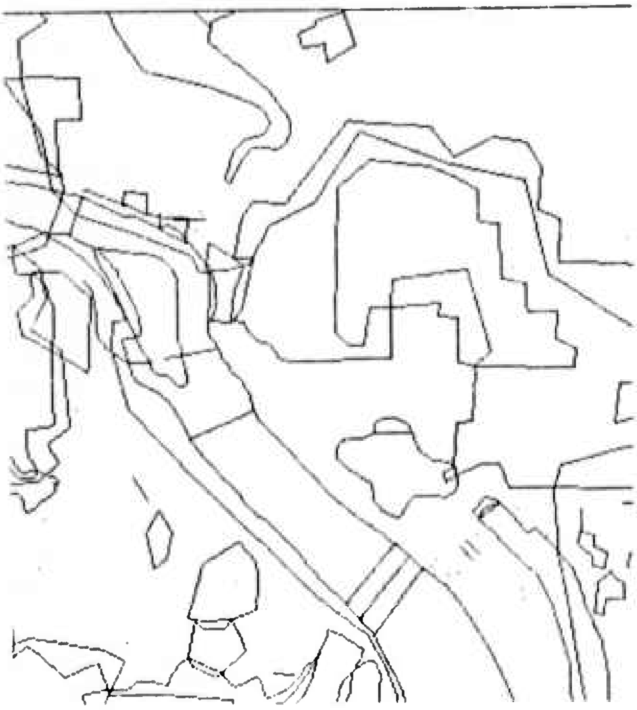


Figure 3: DLMS: Detail of Northwest Washington Area

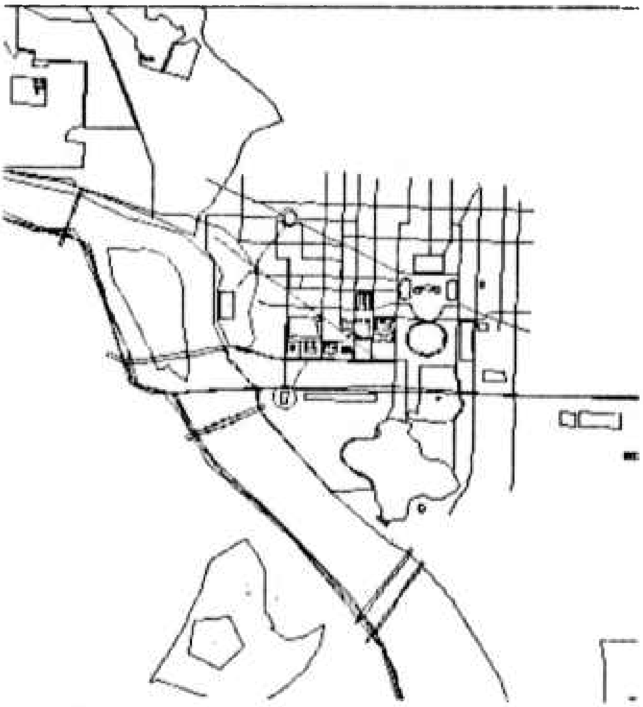


Figure 4: MAPS: CONCEPTMAP Database For Figure3

with DFAD entries primarily because the database was not intended to be used as a general purpose geographic information system. The feature header mechanism allows us to perform random access to features in a map sheet. We can also search using feature attributes such as feature analysis code, feature type, surface material code, and feature id code. This type of reorganization is necessary to support an interactive query-based interface for human and application programs.

Figure 2 shows a plot of polygon features in the area corresponding to our entire Washington D.C. database. Figure 3 is a detailed portion of the DFAD database centered on Foggy Bottom. For comparison, Figure 4 is the corresponding area from the CONCEPTMAP database plotted on the same scale.

Some of the DFAD database entries are easily recognizable as natural or man-made features, although as discussed, this information is not in the original database itself. Figure 5 is the description for the Tidal Basin, Figure 6 is the Rochambeau Bridge. Figure 7 is a description for a large irregular area in central Washington D.C. that contains the major government office buidings. The feature name assigned by MAPS is the first entry in each of the Figures.

```
feature 'd25f471a909'
feature header: 471   (seek:72416)
feature analysis code: 1082
feature type: areal feature
surface material code: (6) water
feature id code: (909) not assigned
subcategory: fresh water (shallow)
average height (meters): 0
aerial feature: 471   polygon with 76 vertices
tree cover: 0   roof cover: 0   density: 0
min point (south west) 5298,7979
max point (north east) 5567,8385
```

Figure 5: DFAD: Description for Tidal Basin

```
feature 'd25f4741250'
feature header: 474   (seek:73132)
feature analysis code: 1085
feature type: linear feature
surface material code: (3) stone / brick
feature id code: (250) not assigned
subcategory: not assigned (general)
average height (meters): 2
linear feature: 474   line with 3 vertices
width: 24   reflectivity: 2
first point: 5024,8064
last  point: 5192,8227
```

Figure 6: DFAD: Description for Rochambeau Bridge

```
feature 'd25f402a610'
feature header: 402   (seek:63688)
feature analysis code: 1010
feature type: areal feature
surface material code: (3) stone / brick
feature id code: (610) not assigned
subcategory: institutional (general)
average height (meters): 28
aerial feature: 402   polygon with 27 vertices
tree cover: 10   roof cover: 70   density: 3
min point (south west) 5705,7971
max point (north east) 6260,8799
```

Figure 7: DFAD: Description for Government Buildings

6.4.2. DTED: Terrain Elevation Database

The organization of the digital terrain database is more straightforward. The DTED database covers the same geodetic area as our DFAD data. It is organized into 64 raster images using the same image format as our digital aerial imagery. Each image containing a 15' x 15' array of terrain samples, where each "pixel" is a discrete elevation point. The terrain package, ELEVATION, provides a transparent interface to the DTED database. Users can retrieve elevation information based on rectangular geodetic area, closest sample point to a geodetic point, or by weighted interpolation. ELEVATION uses the CMU image package to efficiently buffer blocks of contiguous terrain data.

6.5. Conceptual Map Database

The map database component of MAPS, CONCEPTMAP, has been described in McKeown[25]. We will give a brief overview of the organization and concentrate on our new work in hierarchical organization and feature semantics.

6.5.1. Concept Schema

The basic entity in the CONCEPTMAP database is the concept schema. The schema is given a unique ID by the database, and the user specifies a 'symbolic' print name for the concept. Each concept may have one or more role schema associated with it. Role schema specify one or more database views of the same geographic concept. For example, 'northwest washington' can be viewed as a residential area as well as political entity. Another aspect is the ability to associate the same name to two different but related spatial objects. Consider the 'kennedy center' as a building and as the spatial area (ie, lawn, parking area, etc.) encompassing the building. The principle role of a concept schema indicates a preferred or default view. The CONCEPTMAP database is composed of lists of concept schema.

6.5.2. Role Schema

The role schema is a further specification of the attributes of the map feature. It contains the *role name* attribute (building, bridge, commercial area, etc.), a *subrole name* attribute (house, museum, dormitory, etc.), a *role class* attribute (ie., buildings may be *government, residential, commercial,* etc.), a *role type* attribute (ie. physical, conceptual or aggregate), and a *role derivation* attribute (ie. derivation method).

The role name, subrole, and role class attributes categorize the map feature according to its function. For example: this feature is a building, used as an office building, used for government purposes. The role type attribute describes whether the map feature is physically realized in the scene, or if it is a conceptual feature such as a neighborhood, political, or geographic boundary. The role type attribute also provides a mechanism to define the role schema as a collection of physical or conceptual map features. For example, the concept schema in MAPS for 'district of columbia' has a role type aggregrate-conceptual, with aggregrate roles, 'northwest washington', 'northeast washington', 'southwest washington', and 'southeast washington'. This mechanism allows the user to explicitly represent concepts that are strictly composed of other role schema. The role derivation attribute describes the method by which the role and its associated geodetic position description were added to the CONCEPTMAP database.

Each role schema contains a 3DID identifier that is used to access a set of CONCEPTMAP database files which contain geodetic information about the map feature. These identifiers can be shared when multiple roles have the same geodetic description, as in the previous example of 'northwest washington' viewed as both a residential and political area. The CONCEPTMAP 3D description allows for point, line, and polygon features as primitives, and permits the aggregration of primitives into more complex topologies, such as regions with holes, discontinous lines, and point lists. Associated with each feature that was acquired from a image in the database is the generic name of the image. If the correspondence of the generic image changes due to the addition of more ground control points, or better a camera model, the position of the ground feature can be automatically recalculated.

The following is the set of files associated with each 3DID.

- [D3] **3D geodetic location**
  - a set of <latitude/longitude/elevation> triples which define the geodetic position of the role
- [D3F] **3D feature shape description**
  - metric values for lenght, width, area, compactness, centroid, fourier shape approximation etc.
- [FC] **feature image coverage**
  - a list of generic images which contain this feature
  - image mbr and feature coordinates for each image
- [PROP] **feature property list**
  - list of properties of the map feature
  - some general properties such as 'age', 'capacity', '3D display type'
  - feature type specific properties such as 'number of floors', 'basement', 'height', and 'roof type' for buildings

6.5.3. Database Query

CONCEPTMAP supports four methods of database query. The methods are *signal access, symbolic access, template matching* and *geometric access*. The following table gives a brief description of each query method.

- **signal access**
  Given a geodetic specification (point, line, area)*****, perform the following operations:
  - display all imagery at which contains point, line or area.
  - retrieve all map features within geodetic specification
  - retrieve terrain elevation
- **symbolic access**
  Given a symbolic name, such as 'treasury building' perform the following operations:
  - convert name into geodetic specification to perform signal access operations listed above
  - retrieve database description, facts and properties of the map feature
  - retrieve imagery based on symbolic (generic) name
- **template matching**
  Given a partial specification of symbolic attributes perform the following operations:
  - find all map features which satisfy the specification template and return their symbolic name
  - find all images and return symbolic (generic) name
- **geometric access**
  Given a geometic operation such as 'contains' and a geodetic specification perform the following operations:
  - find all map features which satisfy the operation performed over the geodetic specification and return their symbolic name. - find all image features and return symbolic name

These primitive access functions can be combined[25] to answer queries such as: 'display images of Foggy Bottom before 1977', 'what is the closest commercial building to this geographic point', and 'how many bridges cross between Virginia and the District of Columbia'. Figure 8 is a simple schematic giving the processes by which MAPS provides *signal* and *symbolic* access into the CONCEPTMAP database and display of the query result.

6.5.4. Spatial Computation

CONCEPTMAP computes geometric properties based on the geodetic descriptions associated with each role schema in the database. A static description of all spatial relationships between map features for contains, subsumed by, intersection, adjacency, closest point, partitioned by is maintained in the database.

- **'contains'**
  - an unordered list of features which the map feature contains
- **'subsumed by**
  - an unordered list of features which contain the map feature

***** this specification may be in geodetic coordinates or require image-to-map correspondence

- **'intersection'**
  - an unordered list of features which intersect the map feature
- **'closest point'**
  - single feature which is closest to the map feature
- **'adjacency'**
  - an unordered list of features that are within a specific distance of the map feature
- **'partitioned by'**
  - the locus f points where two areal features share a common boundary.

If one or more of the map features in a spatial computation is a result of a dynamic query (and therefore not in the static database), these relationships are computed as needed. A simple 'memo' function is implemented to avoid recomputation of dynamic properties. The use of the static description can also be 'turned off' to evaluate hierarchical search as described in the following section.

The CONCEPTMAP database stores both factual and exact information describing the spatial relationship. For example, if two features intersect, the list of geodetic intersection points is stored, as well as the fact that they intersect at least once. This is necessary for query which require the display of imagery containing a geometric fact, and may possibly be useful for describing the semantics of the intersection. In the following section we will discuss the use of a hierarchical organization based on the 'contains' relation primitive, and show how it can be used to structure the spatial database.

6.6. Hierarchical Organization

In this section we discuss the use of hierarchical organization of spatial data in the MAPS system. The CONCEPTMAP database is used to build a *hierarchy tree* data structure which represents the whole-part relationships and spatial containment of map feature descriptions. This tree is used to improve the speed of spatial computations by constraining search to a portion of the database. In the following sections we briefly discuss why we believe this is a good alternative to regular spatial decompositions such as quadtree[15, 16], or k-d tree[17] usually proposed for MPD model databases.

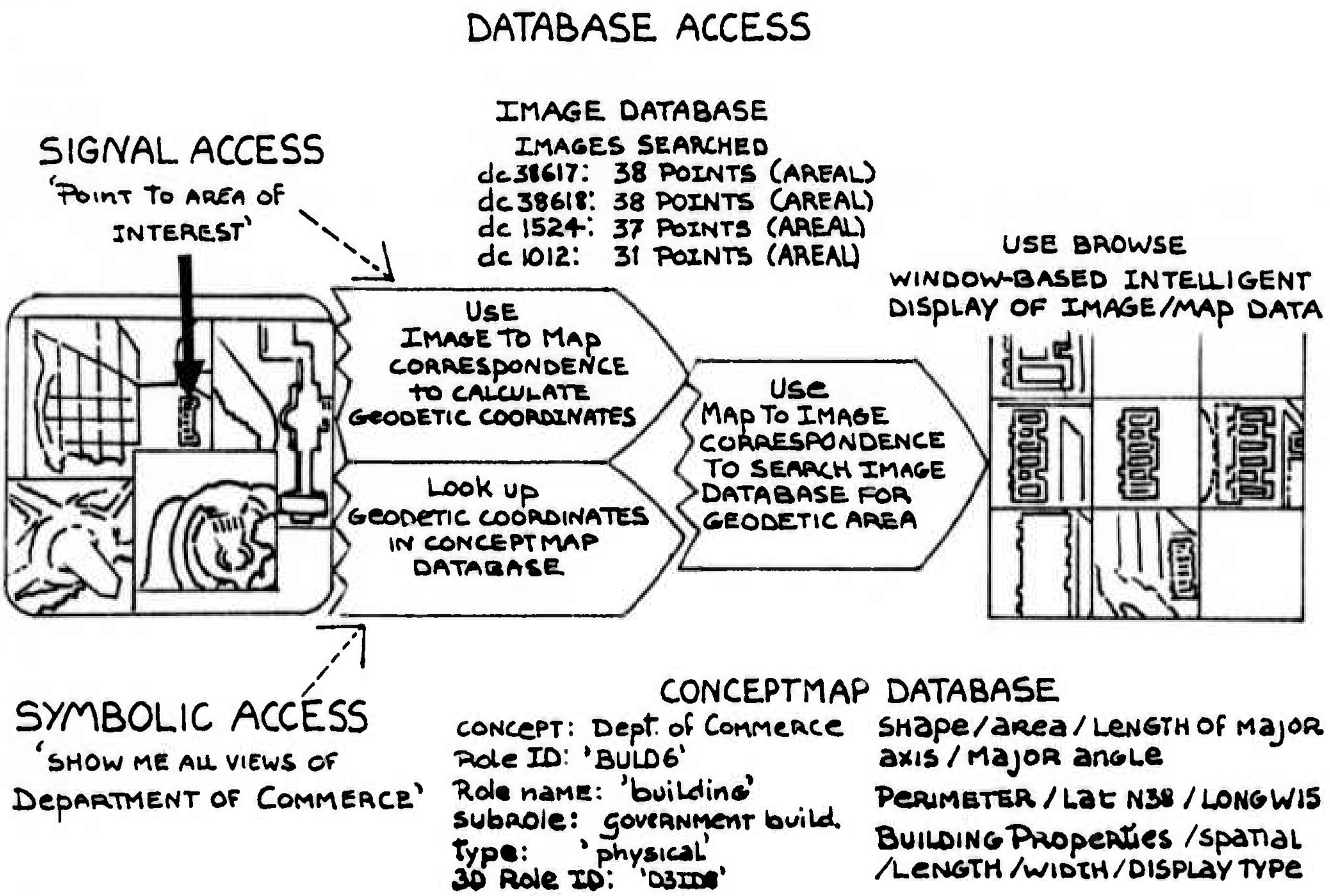

DATABASE ACCESS
IMAGE DATABASE
IMAGES SEARCHED
dc38617: 38 POINTS (AREAL)
dc38618: 38 POINTS (AREAL)
dc 1524: 37 POINTS (AREAL)
dc 1012: 31 POINTS (AREAL)
SIGNAL ACCESS
'POINT TO AREA OF INTEREST'
USE IMAGE TO MAP CORRESPONDENCE TO CALCULATE GEODETIC COORDINATES
LOOK UP GEODETIC COORDINATES IN CONCEPTMAP DATABASE
USE MAP TO IMAGE CORRESPONDENCE TO SEARCH IMAGE DATABASE FOR GEODETIC AREA
USE BROWSE
WINDOW-BASED INTELLIGENT DISPLAY OF IMAGE/MAP DATA
SYMBOLIC ACCESS
'SHOW ME ALL VIEWS OF DEPARTMENT OF COMMERCE'
CONCEPTMAP DATABASE
CONCEPT: Dept. of Commerce
Role ID: 'BULD6'
Role name: 'building'
subrole: government build.
Type: 'physical'
3D Role ID: 'D3ID8'
shape/area/length of major axis/major angle
Perimeter/Lat N38/LongW15
Building Properties/Spatial/Length/Width/Display Type

Figure 8: MAPS: Signal / Symbolic Database Access

6.6.1. Regular Decomposition

Regular decompositions such as the quadtree organizations do not explicitly exploit the inherent structure in spatial organizations. Practical implementations of these organizations often use image-based (integer) coordinate systems and therefore have a bounded position resolution. In general cartographic systems it is important to be able to represent and manipulate map feature descriptions at radically different resolutions using a real valued coordinate system. For example, consider a dynamic query that results in the creation of a very small polygonal area. When computing containment or intersection against a static map database with features represented as a quadtrees, the quadtrees for the static map feature must be generated to a much finer level of detail in order to compare the two data structures. Recent work is beginning to represent quadtrees on real valued coordinate systems[15], but little is known of its practical implementation, complexity, and storage efficiency. K-d trees show storage efficiency improvements over quadtrees[17], since they allow for a more flexible decomposition tailored to spatial feature density. However, they have the same fundamental limitations when used to represent map features in a real valued coordinate system.

In MAPS we perform geometric computations on the feature data in the geodetic coordinate system using point, line, and polygon as map primitives. We constrain search by using a hierarchical representation computed directly from the underlying map data. These spatial contraints can be viewed as natural, that is, intrinsic to the data, and may have some analogy to how humans organize a "map in the head" to avoid search. For example, when a tourist who is looking for the Watergate Hotel is told that the building is in Northwest Washington, she will not spend much time looking at a map of Virginia. Depending on her familiarity with the area, she may avoid looking at much of the map outside of the Northwest District.[******] As we begin to represent large numbers of map features with more complex interrelationships, we believe that the use of natural hierarchies in urban areas, such as political boundaries, neighborhoods, commercial and industrial areas, serve to constrain search. They may also allow us to build systems that organize data using spatial relationships that are close to human spatial models.

6.6.2. Hierarchical Decomposition

The hierarchical containment tree is a tree structure where nodes represent map features. Each node has as its descendants those features that it completely contains in *<latitude/longitude/elevation>* space. The hierarchical tree is initially generated by obtaining an unordered list of features (containment list) for each map database feature. Starting with a designated root node ('greater washington d.c.') which contains all features in the database, descendant nodes are recursively removed from the parent node list if they are already contained in another descendant node. The result is that the parent node is left with a list of descendant features that are not contained by any other node. These descendant nodes form the next level of an N-ary tree ordered by the 'contains' relationship. This procedure is performed recursively for every map feature. Terminal nodes are point and line features, or areal features that contain no other map feature. We will discuss the point containment and closest point computation using the hierarchy tree in the following section.

Figure 9 shows a small section of the hierarchical containment tree. The use of conceptual features-- features with no physical realization in the world but represent well understood spatial areas-- can be used to partition the database. In this case the map feature 'foggy bottom'

```
41 entries for 'contains' for 'northwest washington'
entry 0: 'mcmillan reservoir (role: 0)'
entry 1: 'kennedy center (role: 0)'
entry 2: 'ellipse (role: 0)'
entry 3: 'executive office building (role: 0)'
entry 4: 'white house (role: 0)'
entry 5: 'treasury building (role: 0)'
entry 6: 'department of commerce (role: 0)'
entry 7: 'museum of history and technology (role: 0)'
entry 8: 'key bridge (role: 0)'
entry 9: 'thomas circle (role: 0)'
entry 10: 'dupont circle (role: 0)'
entry 11: 'foggy bottom (role: 0)'
entry 12: 'whitehurst freeway (role: 0)'
entry 13: 'mclean gardens (role: 0)'
entry 14: 'macomb playground (role: 0)'
entry 15: 'theodore roosevelt island (role: 0)'
entry 16: 'interior department (role: 0)'
entry 17: 'district building (role: 0)'
entry 18: 'lafayette park (role: 0)'
entry 19: 'constitution hall (role: 0)'
entry 20: 'national press building (role: 0)'
entry 21: '23rd street (role: 0)'
entry 22: 'constitution avenue (role: 0)'
entry 23: 'virginia avenue (role: 0)'
entry 24: 'national zoo (role: 0)'
entry 25: 'georgetown (role: 0)'
entry 26: 'glover park (role: 0)'
entry 27: 'national cathedral (role: 0)'
entry 28: '21st street (role: 0)'
entry 29: 'north 20th street (role: 0)'
entry 30: '19th street (role: 0)'
entry 31: 'east pennsylvania avenue (role: 0)'
entry 32: 'e street (role: 0)'
entry 33: 'treasury place (role: 0)'
entry 34: 'state place (role: 0)'
entry 35: '26th street (role: 0)'
entry 36: 'west pennsyvania avenue (role: 0)'
entry 37: '16th street (role: 0)'
entry 38: 'l street (role: 0)'
entry 39: 'vermont avenue (role: 0)'
entry 40: '13th street (role: 0)'

11 entries for 'contains' for 'foggy bottom'
entry 0: 'kennedy center (role: 1)'
entry 1: 'washington circle (role: 0)'
entry 2: 'state department (role: 0)'
entry 3: 'american pharmaceutical association (role: 0)'
entry 4: 'national academy of sciences (role: 0)'
entry 5: 'federal reserve board (role: 0)'
entry 6: 'national science foundation (role: 0)'
entry 7: 'civil service commission (role: 0)'
entry 8: 'c street (role: 0)'
entry 9: '22nd street (role: 0)'
entry 10: 'south new hampshire avenue (role: 0)'
```

Figure 9: MAPS: Hierarchical Spatial Containment

[******] If she is told that the Watergate is also near the Potomac river, that should further constrain her search, but that is another story.

allows us to partition some of the buildings and roads that are contained within 'northwest washington'. As more neighborhood areas and city districts are added to our database, we expect to see improved performance especially in areas with dense feature distributions. This will also improve the richness of the spatial description available to the user.

6.6.3. Hierarchical Search

In this section we discuss the use of our hierarchical organization to partition the map database to improve performance by decreasing search when computing the spatial relationships of map features. The hierarchical searching algorithm is basically an N-ary tree searching algorithm. Consider a user at the CONCEPTMAP image display who invokes the geometic database to compute a symbolic description of what map feature he is pointing at. First, using image-to-map correspondence, the system calculates the following map coordinates:

```
latitude  N 38 53 49 (276)
longitude W 77 03 53 (337)
```

This point is converted into a temporary map database feature and is tested against the root node of the hierarchy tree. If it is not contained in this node (not generally the case), then the point cannot correspond to a database feature, and the search terminates. The user is informed that the point is outside the map database.•••••• If the 'contains' test succeeds, it recurses down the tree and performs the test against the siblings of the node just tested. The search allows several paths to exist for any point, thus more than one sibling may contain a path to the point. This sort of anomaly occurs when a feature happens to exist in the intersecting region of two larger regions. However, if the feature is not contained by the node, it is not contained by any of the node's descendants, and that portion of the tree is not further searched. Figure 10 shows the answer to our hypothetical query. The query point is contained within 'theodore roosevelt island', and two search paths in the containment tree are given. The same mechanism is used for line and polygon features, although the primitive determination of containment depends on the geometric type of the feature.

```
This node belongs in the following place(s):
3 entries for 'contains' for 'theodore roosevelt island'
entry 0:          'northwest washington'
entry 1:          'district of columbia'
entry 2:          'greater washington d.c.'
          ••••••••••••• A N D •••••••••••••
2 entries for 'contains' for 'theodore roosevelt island'
entry 0:          'potomac river'
entry 1:          'greater washington d.c.'
```

Figure 10: MAPS: Containment Tree Entry for Theodore Roosevelt Island

6.7. Toward Feature Semantics

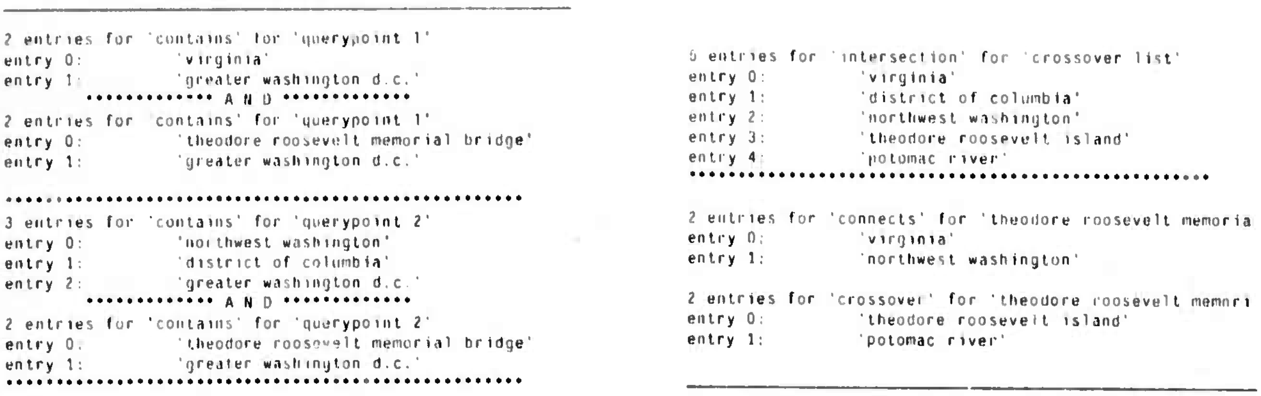
We have begun to investigate the generation of map feature semantics directly from the hierarchical representation of the map feature data. A simple example is the semantic description of a bridge: the feature names and map locations that it connects as well as the names of the map features that it crosses over. Figures 11 and 12 shuw the result of applying a procedural description of the semantics of a bridge concept to calculate the 'connects' and 'crossover' relationship using the map feature descriptions of 'arlington memorial bridge' and 'theodore roosevelt memorial bridge'. These results are generated directly using the MAPS hierarchical organization for spatial data. We do not pose this as a theory of map feature semantics, but envision a set of feature specific procedures that can build these types of descriptions.

```
2 entries for 'contains' for 'querypoint 1'
entry 0:          'virginia'
entry 1:          'greater washington d.c.'
          ••••••••••••• A N D •••••••••••••
2 entries for 'contains' for 'querypoint 1'
entry 0:          'arlington memorial bridge'
entry 1:          'greater washington d.c.'

••••••••••••••••••••••••••••••••••••••••••
4 entries for 'contains' for 'querypoint 2'
entry 0:          'mall area'
entry 1:          'southwest washington'
entry 2:          'district of columbia'
entry 3:          'greater washington d.c.'
          ••••••••••••• A N D •••••••••••••
2 entries for  contains  for 'querypoint 2'
entry 0:          'arlington memorial bridge'
entry 1:          'greater washington d.c.'

••••••••••••••••••••••••••••••••••••••••••
5 entries for 'intersection' for 'crossover'
entry 0:          'virginia'
entry 1:          'district of columbia'
entry 2:          'southwest washington'
entry 3:          'mall area'
entry 4:          'potomac river (Role: 0)'
••••••••••••••••••••••••••••••••••••••••••

2 entries for 'connects' for 'arlington memorial bridge'
entry 0:          'virginia'
entry 1:          'mall area'

1 entries for 'crossover' for 'arlington memorial bridge
entry 0:          'potomac river'
```

Figure 11: MAPS: Semantic Computation from Spatial Data Arlington Memorial Bridge

The procedure for bridge semantics is as follows: A bridge can be represented in the CONCEPTMAP database as an polygonal area, a list of linear segments, or as a geodetic point. The polygonal area arises when the bridge deck is represented, the list of linear segments approximates the center line of the bridge, and the point feature generally represents that the bridge is a landmark feature. No semantics are computed in the latter case. If the bridge is represented as a line, the end points are selected, otherwise the endpoints of the major axis of the bounding ellipse are retrieved from the feature [DB] file. At some level of description, these endpoints define the 'connects' relationship, but this

•••••• This can actually occur since users are allowed to enter arbitrary coordinates thrnugh the terminal. Therefore the database has some crude idea of its extent of map knowledge



```
2 entries for 'contains' for 'querypoint 1'
entry 0:          'virginia'
entry 1:          'greater washington d.c.'
        ••••••••••••• A N D •••••••••••••
2 entries for 'contains' for 'querypoint 1'
entry 0:          'theodore roosevelt memorial bridge'
entry 1:          'greater washington d.c.'

••••••••••••••••••••••••••••••••••••••••••••••
3 entries for 'contains' for 'querypoint 2'
entry 0:          'northwest washington'
entry 1:          'district of columbia'
entry 2:          'greater washington d.c.'
        ••••••••••••• A N D •••••••••••••
2 entries for 'contains' for 'querypoint 2'
entry 0:          'theodore roosevelt memorial bridge'
entry 1:          'greater washington d.c.'
••••••••••••••••••••••••••••••••••••••••••••••
```

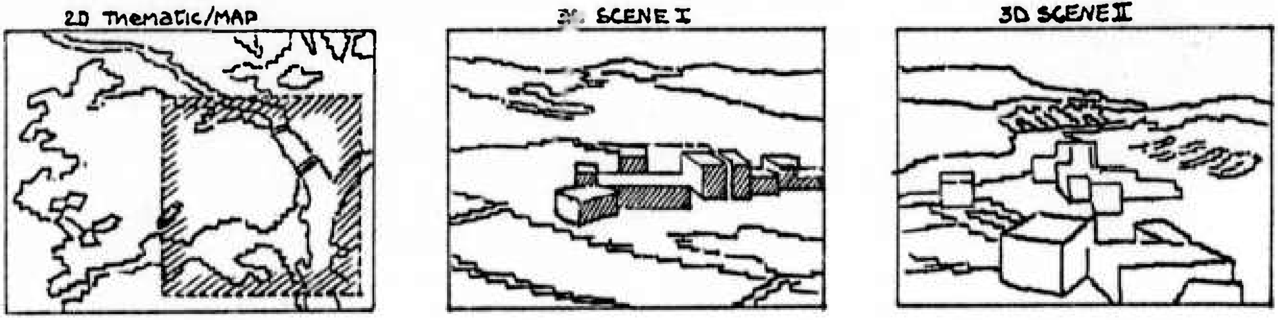
```
5 entries for 'intersection' for 'crossover list'
entry 0:          'virginia'
entry 1:          'district of columbia'
entry 2:          'northwest washington'
entry 3:          'theodore roosevelt island'
entry 4:          'potomac river'
••••••••••••••••••••••••••••••••••••••••••••••

2 entries for 'connects' for 'theodore roosevelt memoria
entry 0:          'virginia'
entry 1:          'northwest washington'

2 entries for 'crossover' for 'theodore roosevelt memori
entry 0:          'theodore roosevelt island'
entry 1:          'potomac river'
```

Figure 12: MAPS: Semantic Computation from Spatial Data
Theodore Roosevelt Memorial Bridge

3D MAP DISPLAY

MAPS can be used to generate a 3D scene of a designated area by combining terrain, conceptmap database and thematic map data

USER specifies area of interest by symbolic or signal access, also specifies 3D viewing position and illumination position



Displays thematic color data

▨	Metal
▦	stone/brick
▤	water
▥	composition
▩	earthen works
▓	trees
□	buildings

searches terrain database to elevation

<W ^N

15	15	15	15	15	15
13	13	14	15	14	13
12	12	12	13	12	12
11	11	11	11	11	11
9	9	10	10	10	10
8	8	9	9	9	8

LAT:	N38	53	21	(499)
LON:	W77	2	8	(149)

searches conceptmap database

Political	
CONCEPT2	district of columbia
CONCEPT3	NORTHWEST WASHINGTON
Natural Features	
CONCEPT1	tidal basin
CONCEPT43	anacostia
Residential	
CONCEPT51	georgetown
CONCEPT 54	buleith

Figure 13: WASH3D: 3D Map Display

is not useful if we are envisioning generation of a reasonably complex symbolic representation.

The 'contains' relationship is applied to each endpoint using the hierarchical tree to order the search. As before, this search returns a list of features ordered by spatial containment, and there may be several independent containment paths. Redundant paths are eliminated by examining whether the bridge is in the containment path. The first entry (0) in each of the remaining paths is one of the areas connected by the bridge. Using the 'contains' relationship, the other entries in the path are also valid connecting areas.

To compute the 'crossover' relationship, the 'intersection' relationship is computed for the bridge using the complete list of line segments or the polygonal description. A list of all the features that the bridge intersects is assembled. Entries in the intersection list are removed if they are also present in either of the 'connects' lists. The assumption is that those features that didn't contain a bridge endpoint, but intersected with the bridge description, are those features that the bridge crosses over. If there is sufficiently detailed elevation data for man-made features it should be possible to compute semantics for 'passes over' and 'passes under' by calculating the feature elevation at the actual geodetic point of intersection.

7. Synthesis Tasks

In this section we will discuss three applications of the MAPS database to cartographic and image interpretation tasks. These tasks are 3D scene generation of views of Washington D. C., the use of the map database to guide image segmentation, and some preliminary results on a rule-based system for airport scene interpretation. Each task requires the capabilities of various aspects of the IMD model as implemented in the MAPS system. These applications pull together external and image/map databases, and are only possible using an integrated system that relates imagery, terrain, and map data through a unified cartographic representation.

7.0.1. WASH3D: 3D Scene Generation

The first application of the MAPS database is in the area of 3D computer graphics for scene simulation and database validation. Computer graphics play an important role in the areas of image processing, photo-interpretation, and cartography. In cartography various phases of the map generation process use graphics techniques or source material analysis, transcription and update, and some aspects of map layout and production. However, many major steps in the generation of a cartographic product remain largely manual. One important step for which inadequate tools exist is the integration of terrain and cultural feature databases. This integration step is often

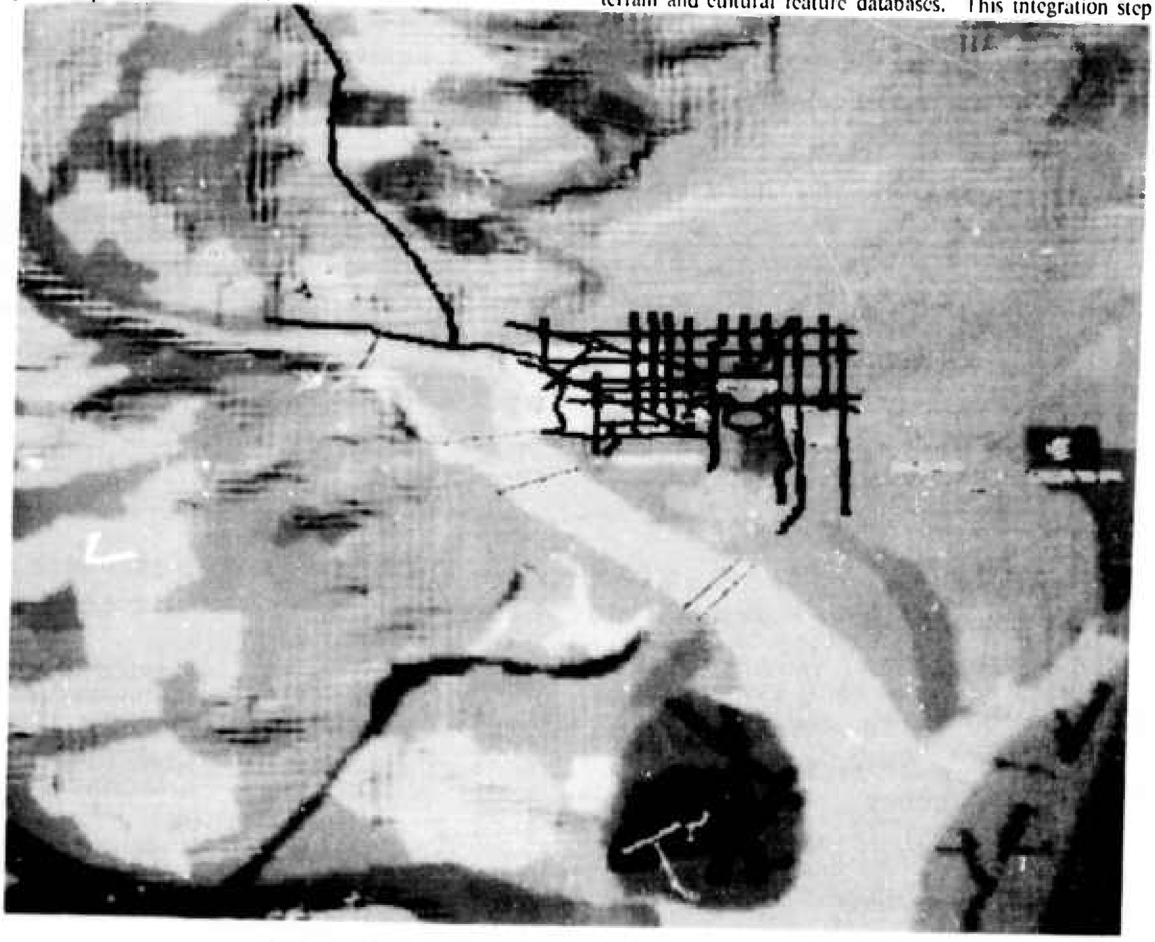


Figure 14: WASH3D: Vertical View 85^0 Northwest Washington

used to verify the geodetic accuracy of natural and man-made features in the digital database prior to actual map layout and production. Another application is sensor simulation[30, 31]. Radar, visual, and multi-sensor scenes are digitally generated to verify the quality of digital culture and terrain databases or to determine the quality of the sensor model. Improvements to the level of detail contained in the underlying database can be subjectively measured in terms of the quality of the generated scene.

WASH3D[32] is an interactive graphics system that uses the MAPS system to integrate a digital terrain database, a cultural feature database, and the CONCEPTMAP database to allow a user to generate cartographically accurate 3D scenes for human visual analysis. WASH3D uses the coarse resolution DLMS database described in Section 6.4 to generate a baseline thematic map. The thematic map is a 2D image which is produced by scan conversion of the DLMS digital feature analysis database (DFAD) polygon database. We assign a color to each region polygon using the DFAD surface material code-- forest and park (green), water (blue), residential (yellow), and high-density urban (brown). DLMS terrain elevation data (DTED) is interpolated to determine ground elevations at each point in the 2D image. Since the resolution of the DFAD data is coarse, comparable to map scales of 1:250,000 to 1:100,000, we use the CONCEPTMAP database to provide high resolution 3D feature descriptions of buildings, roads, bridges, residential and commercial areas. The CONCEPTMAP database is derived from imagery with resolutions between 1:12000 and 1:36000, and the addition of these features effectively intensifies the perceived level of detail in the simulated scene, even though the base map is at a coarse resolution.

Lukes[33] describes the utility of selective database intensification for tailoring standard database products to custom applications and for time-critical applications which cannot be handled by normal production schedules. Figure 13 shows the interactive process by which users can specify an area of interest for 3D scene generation. Figures 14 and 15 show two 3D scenes of the Washington D.C. area generated by WASH3D.

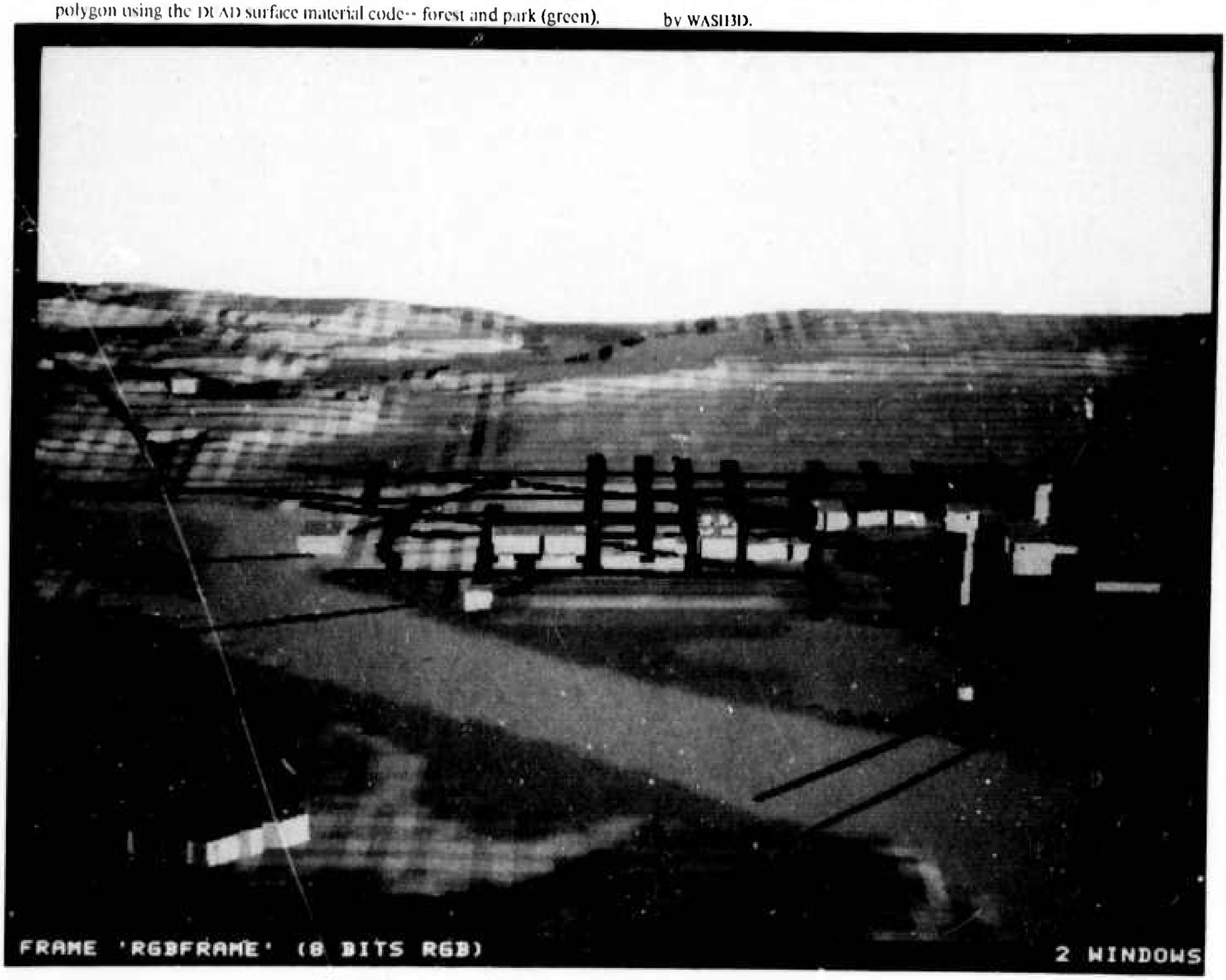


Figure 15: WASH3D: Northwest Washington From Above National Airport

7.0.2. MACHINESEG: Map-Guided Machine Segmentation

The second application of the MAPS database is in the area of map-guided machine segmentation. Users may specify a map feature from the CONCEPTMAP database or interactively generate a feature description using the SEGMENT program. In the case of a map database feature, MACHINESEG uses an existing image coverage [IC] file (see Section 6.5.2) that specifies in which images the feature is found, and the feature location in the image. For interactive specification, an [IC] file is created dynamically by image-to-map correspondence using the image database.

For each image, a high resolution window containing the database feature is extracted and displayed. We expand the size of the image window to contain an area of uncertainty around the feature location. The expansion is currently based on the size of the feature, but we plan to incorporate correspondence error measures based on the quality of the camera model associated with each image. The image window is smoothed, and a segmentation is performed using a region-growing technique[34] which combines an edge strength metric and region merge acceptability based on spectral similarity to control region growing.

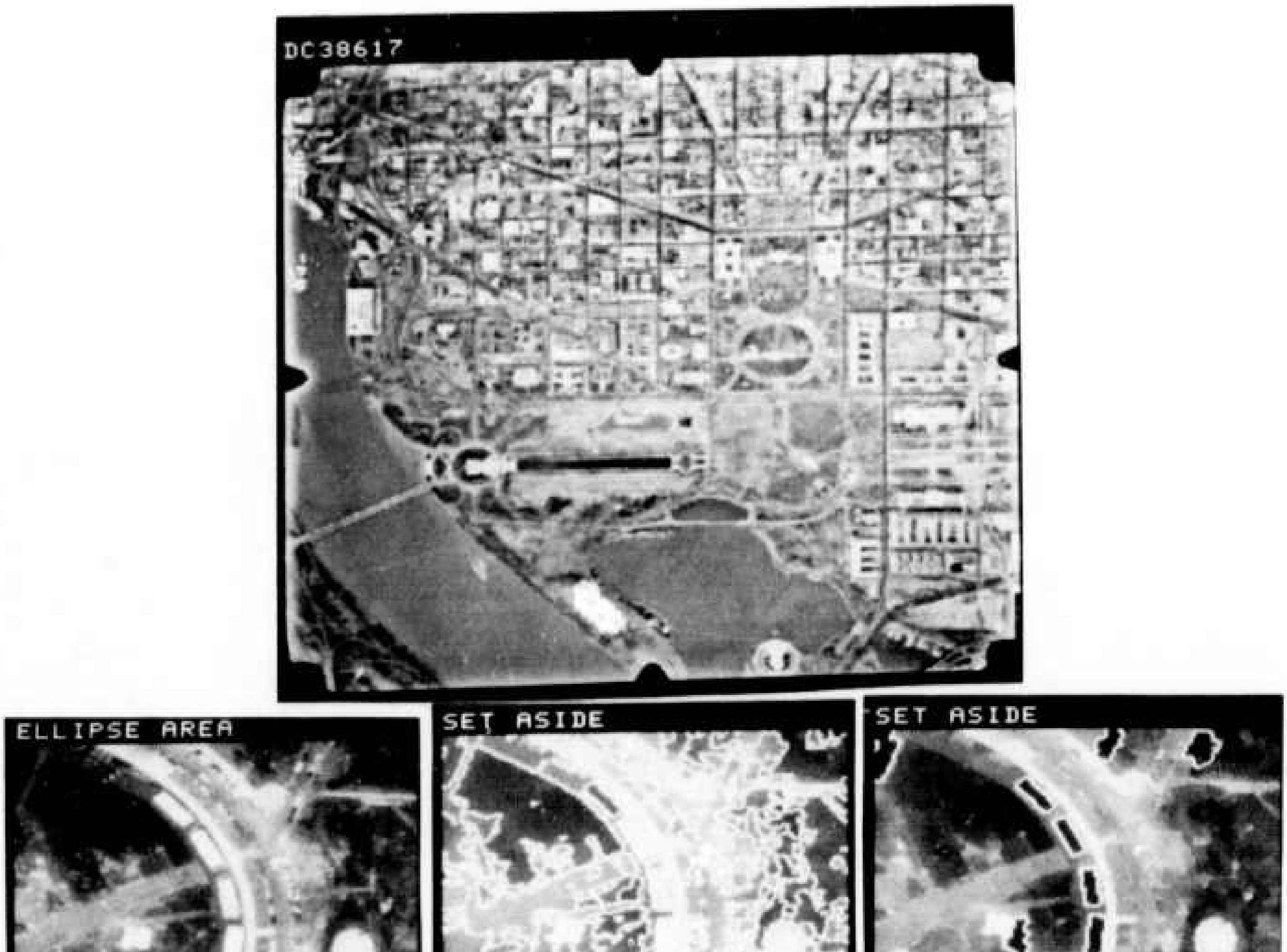


Figure 16: MACHINESEG: Segmentation using MAPS System

Figure 16 shows the segmentation of several low-elevation buildings along the perimeter of the Washington Ellipse. The uppermost building is added to the CONCEPTMAP database in the standard manner described in Section 6.5. The user specifies the image, DC38617, to perform the segmentation and the MACHINESEG system automatically displays a reduced resolution window of the image (*dc38617*), and a high resolution window (*ellipse area*) containing the database area. MACHINESEG creates a copy of the high resolution window as a work area (*set aside*) for the image processing routines. An image smoothing operation is followed by the generation of seed regions using a conservative similarity measure to insure that potentially matchable regions are not prematurely merged. The initial seed regions are overlaid on the image using graphics overlays. Any seed regions that satisfy the shape criteria for the database feature are extracted and marked. In this example, the database feature itself was marked in the initial seed region matching. As regions are merged based on weak edge boundaries and high spectral compatibility, the resulting region is evaluated with respect to a list of shape and spectral criteria. If the region satisfies the criteria, it is marked, and further merging is allowed only if the proposed merge improves the overall region score. Criteria include fractional fill, area, linearity, perimeter, compactness, and spectral measures.

The final results are shown in the second window labeled *set aside*. Five buildings similar to the map database feature were correctly identified while one building was omitted. Six segments were incorrectly identified. Had we made use of spectral information in this particular segmentation-- that the building roofs were bright features-- we probably could have excluded 5 of the 6 errors. However, we are more concerned with using weak knowledge, and one cannot expect better performance without more sophisticated analysis. MACHINESEG allows the user to delete erroneous segments and generates map descriptions of each extracted feature. These descriptions can then be used to search for these features in other database imagery.

The significance of MACHINESEG is that it can search systematically for features in a database of images, an operation that is fundamental for change detection applications. It directly uses the map database description as an evaluation tool for image segmentation and interpretation. It also uses very general image processing tools to perform both segmentation and evaluation and is amenable to supporting other approaches to image segmentation and feature recovery. A further application of the MACHINESEG system is discussed in the following section.

7.0.3. SPAM: Rule-based System for Airport Interpretation

The third application of the MAPS system is in the investigation of rule-based systems for the control of image processing and interpretation with respect to a world model.

In photo-interpretation, knowledge can range from stereotypical information about man-made and natural features found in various situations (airports, manufacturing, industrial installations, power plants etc.) to particular instantiations of these situations in frequently monitored sites. It is crucial for photo-interpretation applications that the metrics used be defined in a cartographic coordinate system, such as *<latitude/longitude/elevation>*, rather than an image-based coordinate system. Descriptions such as "the runway has area 12000 pixels" or "houses are between 212 and 345 pixels" are useless except for (perhaps) the analysis of one image. It is the case, however, that to operationalize metric knowledge one must relate the world model to the image under analysis. This should be done through image-to-map correspondence using camera models which is the method used in our system.

We have begun to build SPAM[35] to test our ideas in the use of the combination of a map database, task independent low-level image processing tools, and a rule-based system.

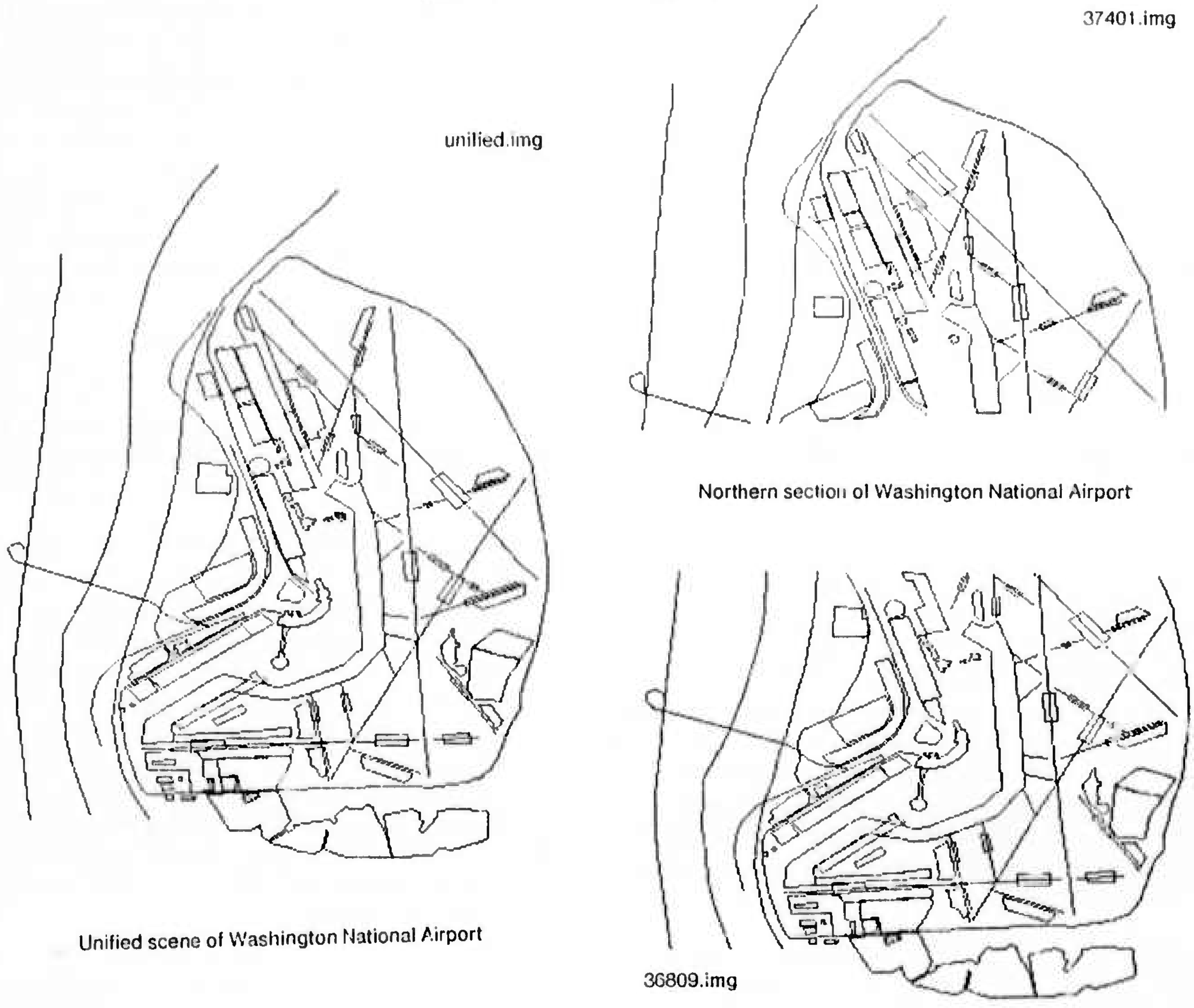
SPAM uses the MAPS database to store facts about man-made or natural feature existence and location, and to perform geometric computation in *map space* rather than *image space*. Differences in scale, orientation, and viewpoint can be handled in a consistent manner using a simple camera model. The MAPS database facility also maintains a partial model of interpretation, separate from, but in the same representation as, the map feature database.

The image processing component is based on the MACHINESEG program described in the previous section. It performs low level and intermediate level feature extraction. Processing primitives are based on linear feature extraction and region extraction using edge-based and region-growing techniques. It identifies islands of interest and extends those islands constrained by the geometric model provided by MAPS and model-based goals established by the rule-based component.

The rule-based component provides the image processing system with the best next task based on the strength/promise of expectations and with constraints from the image/map database system. It also guides the scene interpretation by generating successively more specific expectations based on image processing results.

We are in the preliminary stages of development for the SPAM system and have begun to build a detailed map model of National Airport. Figure 17 gives an example of the ability of the MAPS database to use image-to-map correspondence to generate unified spatial models from partial information. The line drawing labeled 37401 IMG contains the northern section of National Airport; 36809 IMG is a partially overlapping southern section of National Airport. Line segments represent point, line, and areal features corresponding to runways, terminal buildings, access roads, and hangars, interactively specified

Figure 17: SPAM: National Airport Spatial Model



unified.img

Unified scene of Washington National Airport

37401.img

Northern section of Washington National Airport

36809.img

Southern section of Washington National Airport

using the CONCEPTMAP representation. For those features that appear in both images, the concept *role* mechanism (see Section 6.5.2) is used to specify multiple *<latitude/longitude/elevation>* descriptions. A unified map description is created by matching corresponding line segments using the overlapping image areas (in map space) to constrain search. The result of unification is the line drawing labeled AIRPORT.IMG.

8. Future Work

Our future work will be directed toward two research topics. First, we have only begun to explore the use of MAPS as a component of an image interpretation system. We will continue our work in the airport scene interpretation task, using the SPAM system as a testbed for integration of a rule-based system with the MAPS system. Second, there is much to do in expanding the CONCEPTMAP database to include more complex 3D descriptions, and in attendant issues of scaling and sizing to larger databases. Other tasks we will pursue are the evaluation of our hierarchical spatial representation to constrain search in large databases, general solutions to complex spatial queries for situation assessment applications, and the application of spatial knowledge to navigate through a map database.

In discussing future work it is important to understand the strengths and limitations of the current research. The strengths of this work lie in several unique features of the MAPS system. First, we have constructed a system of moderate complexity which has significant capabilites in each area of our Image/Map Database model. The system integrates map knowledge from diverse sources and performs several tasks that require synthesis of this knowledge. We have the ability to represent complex map features in a uniform cartographic coordinate system and can compute new spatial relationships directly from the map data.

The major limitation in the MAPS system is the current method for performing image-to-map correspondence.[•••••••] From the standpoint of the state of the art in photogrammetry, we make simplistic planemetric assumptions in our correspondence algorithm, but they do give reasonable results for several reasons. First, all of our photographs are vertical aerial mapping imagery, and efforts are taken to minimize camera tilt. Second, we have very high resolution photographs, each of which covers a relatively small area, and due to the relatively local level terrain in Washington D. C., our polynomial correspondence functions are reasonably accurate.

The issue is not how to recover camera information from the imagery, since in cartography and manual photo-interpretation the sensor models and ephemeral data are well known and modeled, but to use existing photogrammetric tools for basic data acquisition. Therefore, in this limitation we see an opportunity to investigate how MAPS could be interfaced to a photogrammetric frontend which would directly provide *<latitude/longitude/elevation>* data from a stereo model.[•••••••••] The frontend should have a landmark database and interactive display tools to guide the stereo model setup in a manner similar to our current implementation. Nothing in the current MAPS implementation precludes such an interface since we maintain a 3D map feature representation throughout the database using the USGS terrain database. The building of such tools should be the common objective both to cartographers and to computer scientists.

9. Acknowledgements

The MAPS system is a result of effort by several people. Jerry Denlinger implemented the BROWSE system implementation. Wilson Harvey implemented the hierarchical tree generation and supervised much of the landmark database generation. Mike Matsko implemented WASH3D, the 3D scene generation system. Dave Springer is responsible for image database generation and maintenance. Takeo Kanade and Raj Reddy have provided encouragement, support, and guidance for this work.

References

1. McKeown, D. M., Jr. and Reddy, D. R., "A Hierarchical Symbolic Representation for an Image Database," *Proceedings of IEEE Workshop on Picture Data Description and Management*, April 1977, pp. 40-44.

2. McKeown, D. M., Jr., and Reddy, D. R., "The MIDAS Sensor Database and its Use in Performance Evaluation," *Proceedings ARPA Image Understanding Workshop*, Palo Alto, Cal., Oct. 1977, pp. 47-51.

3. Rubin, S. M., *The ARGOS Image Understanding System*, PhD dissertation, Carnegie-Mellon University, Computer Science Department, November 1978.

4. Rubin, S. M., "Natural Scene Recognition Using LOCUS Search," *Computer Graphics and Image Processing*, Vol. 13, 1980, pp. 298-333.

5. Barrow, H., et. al., "Interactive Aids for Cartography and Photo Interpretation: Progress Report, October 1977," *Proceedings: DARPA Image Understanding Workshop*, Oct. 1977, pp. 111-127.

6. Fischler, M., et. al., "The SRI Road Expert: An Overview," *Proceedings: DARPA Image Understanding Workshop*, Nov. 1978, pp. 13-19.

7. Quam, L., "Road Tracking and Anomaly Detection," *Proceedings: DARPA Image Understanding Workshop*, May 1978, pp. 51-55.

8. Nagy, G., and Wagle, S., "Geographic Data Processing," *Computing Surveys*, Vol. 11, No. 2, June 1979, pp. 139-182.

9. *IEEE Workshop on Picture Data Description and Management*, IEEE Computer Society, Asilomar, Ca., August 1980.

10. *IEEE Computer Magazine, Special Issue on Pictorial Information Systems*, IEEE Computer Society, November 1981.

11. Kondo, T., Shinoda, H. Sawada, N., Numagami, H. and Kidode, M., "A Map-Guided Image Database System For Remotely Sensed Data," *Proceedings of Pattern Recognition and Image Processing Conference*, June 1982.

12. Chang, N. S., and Fu, K. S., "A Query Language For Relational Image Database Systems," *Proceedings of the Workshop on Picture Data Description and Management*, IEEE, August 1980, pp. 68-73.

13. Chang, N. S., and Fu, K. S., "An Integrated Image Analysis and Image Database Management System," Tech. report TR-EE-80-20, Electrical Engineering Department, Purdue University, May 1980.

14. Chang, N. S., *Image Analysis and Image Database Management*, UMI Research Press, Ann Arbor, Michigan, 1981.

15. Samet, H., and Webber, R. E., "Using Quadtrees to Represent Polygonal Maps," *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, June 1983, to appear

16. Rosenfeld, A., Samet, H., Shaffer, C., and Webber, R. E., "Application of Hierarchical Data Structures to Geographical Information Systems," Tech. report TR-1197, University of Maryland, 1983.

[•••••••] For tasks such as photo-interpretation and situation assessment, the current level of accuracy using high resolution photographs is not as critical and may be sufficient.

[•••••••••] The CAPIR system[33] is one example of such a system.

17. Matsuyama, T., Hao, L.V., and Nagao, M., "A File Organization for Geographic Information Systems," *Proceedings of 6ICPR*, IEEE, Oct 1982, pp. 83-87.

18. Bullock, B.L., et. al., "Image Understanding Application Project: Status Report," *Proceedings: DARPA Image Understanding Workshop*, Sept. 1982, pp. 29-41.

19. Brooks, R. A., "Symbolic Reasoning among 3-D Models and 2-D Images," *Artificial Intelligence*, Vol. 17, 1981, pp. 285-349.

20. Matsuyama, T., "A Structural Analysis of Complex Aerial Photographs," Tech. report, Department of Electrical Engineering, April 1980, Ph.D Thesis

21. Nagao, M., Matsuyama, T., and Mori, H., "Structural Analysis of Complex Aerial Photographs," *Proceedings of Sixth International Conference on Artificial Intelligence*, IJCAI, Tokyo, Japan, August 1979, pp. 610-616.

22. Selfridge, P. G.., "Reasoning About Success and Failure in Aerial Image Understanding," Tech. report 103, University of Rochester, May 1982, Ph.D Thesis

23. Herman, M., Kanade, T. and Kuroe, S., "Incremental Acquisition of a Three-dimensional Scene Model from Images," Tech. report CMU-CS-82-139, Computer Science Department, Carnegie-Mellon University, October 1982.

24. McKeown, D. M. and T. Kanade, "Database Support for Automated Photo Interpretation," *Techniques and Applications of Image Understanding III*, Society of Photo-Optical Instrumentation Engineers, Washington, D.C., April 1981, pp. 192-198.

25. McKeown, D.M., "Concept Maps," *Proceedings: DARPA Image Understanding Workshop*, Sept. 1982, pp. 142-153, Also available as Technical Report CMU-CS-83-117

26. McKeown, D. M., and J. L. Denlinger, "Graphical Tools for Interactive Image Interpretation," *Computer Graphics*, Vol. 16, No. 3, July 1982, pp. 189-198.

27. Teitelman, W., "A Display Oriented Programmer's Assistant," Tech. report CSL77-3, Xerox Palo Alto Research Center, 1977.

28. Sproull, R. F., "Raster Graphics for Interactive Programming Environments," Tech. report CSL79-6, Xerox Palo Alto Research Center, 1979.

29. Defense Mapping Agency, *Product Specifications for Digital Landmass System (DLMS) Database*, St. Louis, Missouri, 1977.

30. Faintich, M. B., "Digital Sensor Simulation at the Defense Mapping Agency Aerospace Center," *Proceedings of the National Aerospace and Electronics Conference (NAECON)*, May 1979, pp. 1242-1246.

31. Faintich, M. B., "Sensor Image Simulator Application Studies," *Proceedings: International Conference on Simulators*, Sept. 1983, to appear

32. McKeown, D. M., and Matsko, M. S., "Urban Scene Generation for Cartographic Applications," Tech. report, Carnegie-Mellon University, in preparation, 1983.

33. Lukes, G. E., "Computer-assisted photo interpretation research at United States Army Engineer Topographic Laboratories (USAETL)," *Techniques and Applications of Image Understanding III*, Society of Photo-Optical Instrumentation Engineers, Washington, D.C., April 1981, pp. 85-94.

34. Yoram Yakimovsky, "Boundary and Object Detection in Real World Images," *Journal of the ACM*, Vol. 23, No. 4, 1976, .

35. McKeown, D.M., and McDermott, J., "Toward Expert Systems for Photo Interpretation," *IEEE Trends and Applications '83*, May 1983.

MAPS System Major Components

This Appendix contains a list of the major program modules which compose the MAPS system.

NAME	SIZE (bytes)	COMMENTS
Browse		
browse	305500	interactive image display facility
picpac	530762	interactive image processing facility
Corres		
corres	286042	interactive image-map correspondence
checkcorres	49523	check correspondence errors
cormain	52893	correspondence algorithm
corpairs	75176	edit correspondence pairs file
creatsdf	50601	create a scene description file
dumpcoef	19649	dump a coefficients file
dumpcor	23547	dump a correspondence file
dumpsdf	25398	dump a scene description file
hypcorpairs	82380	generate hypothesized landmarks
updatesdf	59099	update a scene description file
Landmark		
landmark	194953	interactive landmark extraction
creatldm	23557	create binary landmark file
etytod3	50217	make a .d3 file frum an .ety file
etytoldm	19948	create landmark file from .ety files
ldescribe	43695	give landmark descriptions
ldmriprt	30275	dump all info about a landmark
ldmtest	28696	find landmarks within geodetic area
Segment		
segment	170230	hand segmentation program
mkidf	10537	create ascii file from binary seg file
segrename	39045	edit segmentation region names
Machineseg		
machineseg	290222	machine segmentation program
Conceptmap		
conceptmap	665710	associate conceptual and map data
buildsegmap	98301	build composite segmentations
coetrack	125241	track points using map correspondenco
congeoall	213278	generate geometric database
d3dump	24629	dump a d3 file
d3entcor	93936	create corres entry from .d3 file
d3fdump	31039	dump a d3 feature file
d3tod3f	15826	convert a .d3 file to a feature file
d3toimg	44710	generate binary image from .d3 files
dlmsseg	128324	create DLMS overlay for geodetic area
dmaextract	31544	extract features from DLMS .fea files
dumpql	207962	dump a querylist file
dumpsdf	25398	dump a scene description file
ocdump	9425	dump the contents of a coverage file
ecshow	137700	display manager for coverage files
ecsort	26624	sort coverage files by keys
ectoseg	18173	create .seg file from coverage file
hierarchy	486262	build and access hierarchical database
hiertrack	321869	track and display pts using hierarchy
idhier	254739	identify points using hierarchy
imagetoec	34283	associate image with coverage file
imagetomap	54092	<generic><row><col> => <lat/lon/elev>
photo	299710	interactive image photogrammetry
segtod3	57034	convert .seg file to .d3 data structure
segtoimg	32785	convert .seg regions to binary image
stereoshow	153125	show stereo image pairs
unifyseg	107603	unify segmentation regions
Wash3d		
wash3d	764517	3d scene generation from MAPS database
dfeaprt	45013	print DLMS feature given dlms code
dispfea	137335	display a DLMS map feature file
dlms	53134	create dlms index file
dlmsbin	34604	convert ascii feature files to binary
dlmsfind	45419	find a DLMS feature hased on attributes
feadumper	45267	dump a DLMS feature file
Terrain		
elevation	24097	access terrain data images

SEGMENT-BASED STEREO MATCHING*

By

Gerard G. Medioni and Ramakant Nevatia

Intelligent Systems Group
University of Southern California
Los Angeles, California 90089-0272

The author note

Abstract

Images are two dimensional projections of three dimensional scenes, therefore depth recovery is a crucial problem in Image Understanding, with applications in passive navigation, cartography, surveillance, and industrial robotics. Stereo analysis provides a more direct quantitative depth evaluation than techniques such as shape from shading, and its being passive makes it more applicable than active range finding imagery by laser or radar. This paper addresses the subproblem of identifying corresponding points in the two images. The primitives we are using are groups of collinear connected edge points called segments, and we base the correspondence on the minimum "differential disparity" criterion. The result of this processing is a sparse array disparity map of the analyzed scene.

I. Introduction

The human visual system perceives depth with no apparent effort and very few mistakes, but how it does so is not understood. Binocular stereopsis plays a key role in this process, and the straightforward extraction of depth it provides, once corresponding points are identified, makes it very attractive. Depth recovery is necessary in domains such as passive navigation[Gennery80, Moravec80], cartography[Kelly77, Panton78], surveillance[Henderson79] and industrial robotics. Proposed solutions for the stereo problem follow a paradigm involving the following steps[Barnard82]:

-image acquisition,
-camera modeling,
-feature acquisition,
-image matching,
-depth determination,
-interpolation.

The hardest step is image matching, that is identifying corresponding points in two images, and this paper is solely devoted to it. The next section reviews the existing systems that have been proposed so far, divided in two broad classes, area-based and edge-based, then we summarize our assumptions and give a formal description of the method. The fourth section presents results, and we then discuss extensions.

II. Review of existing methods

Two classes of techniques have been used for stereo matching, area-based and feature-based.

2.1. Area-based stereo

Ideally, one would like to find a corresponding pixel for each pixel in each image of a stereo pair, but the semantic information conveyed by a single pixel is too low to resolve ambiguous matches, therefore we have to consider an area or neighborhood around each pixel, and use correlation-based matching algorithms to determine the corresponding match, it is therefore using local context to resolve ambiguities. The justification for such an approach is that of "continuity", that is disparity values change smoothly, except at a few depth discontinuities. All systems based on area-correlation suffer from the same limitations:

- They require the presence of a detectable texture within each correlation window, therefore they tend to fail in featureless or repetitive texture environments.
- They tend to be confused by the presence of a surface discontinuity in a correlation window.
- They are sensitive to absolute intensity, contrast and illumination.
- They get confused in rapidly changing depth fields (vegetation.)

For these reasons, the existing systems, specially the ones used in "automatic" cartography, require the intervention of human operators to guide them and correct them. Such systems are described in [Lucas81, Panton78, Hannah80, Barnard80, Moravec79].

*This research was supported by the Defense Advanced Research Projects Agency and was monitored by the Air Force Wright Aeronautical Laboratories under Contract No. F33615-82-K-1786, DARPA Order No. 3119.

2.2. Feature-based systems

The depth information in stereo analysis is conveyed by the differences in the two images of a stereo pair due to the different viewpoints, the differences being most prominent at the discontinuities, or edges. Obviously, matching of features will not provide a full depth map, and must be followed by an interpolating scheme. The common characteristics of feature-based matching techniques are:

- They are faster than area-based methods, because there are many fewer points to consider.
- The obtained match is more accurate, edges can even be located with sub-pixel precision[Binford81].
- They are less sensitive to photometric variations, since they represent geometric properties of a scene.

Henderson[Henderson79] considered scenes representing cultural sites (man-made structures) and matched edge points on epipolar lines in the two views. He reduced ambiguity by assuming continuity between consecutive epipolar lines. Marr and Poggio have relied on two apparently simple constraints[Marr79]:

1. Uniqueness.
   Each point in an image may be assigned at most one disparity value. One may note that this assumption is not correct for transparent objects.
2. Continuity.
   Matter is cohesive, therefore values change smoothly, except at a few depth discontinuities.

They first proposed a cooperative algorithm[Marr76] that works very well on random-dot stereograms, but they rejected it to propose one of more heuristic nature, implemented by Grimson[Grimson79, Grimson81] that generates good results, given the very few assumptions. Arnold[Arnold78] matches edges using local context, and his system seems to perform well on cultural scenes. Finally, Baker and Binford[Baker82] match edges on epipolar lines by using the no-reversal constraint that the order of the match has to be preserved, in addition to uniqueness and continuity. They also consider continuity by examining adjacent epipolar lines. This system appears to perform reasonably on a wide variety of images.

In most of the systems presented above, a considerable saving in search time is obtained by a coarse to fine matching, that is the matching is originally done on a low-resolution version of the image and the results are propagated to the higher resolution version. However, it should be noted that in current implementations, good matches as well as errors tend to propagate from one level to the next.

III. The Minimal Differential Disparity Algorithm

From the survey conducted above, it appears that feature-based techniques are more appropriate to solve the correspondence problem, but edges as a primitive seem to be too low-level, and a connectivity check is needed to remove spurious matches. High level primitives such as physical object boundaries or surface descriptions would be preferred, however, stereo processing may need to precede the computation of such descriptions. As a step towards higher level primitives, we are using segments. In order to generate them, we fit straight lines through adjacent edge points with a given tolerance of one pixel. These segments can be described by:

- coordinates of the end points
- orientation
- strength (average contrast)

By using these primitives, we implicitly assume the connectivity constraint. When matching segments, we need to allow one segment to possibly match with more than one segment in the other image (i.e. to allow for fragmented segments), even if we wish to preserve unique matches for the individual edge points. Also, instead of considering one epipolar line at a time, we have to consider all epipolar lines in which a given segment appears.

3.1. Assumptions and Definitions

We consider a simple camera geometry in which the epipolar plane, defined as the plane passing through an object point and the two camera foci, intersects the two image planes, so defining epipolar lines parallel to the y axis. Therefore, corresponding points must lie on corresponding epipolar lines, that is have the same row value, this is illustrated in Figure 3-1.

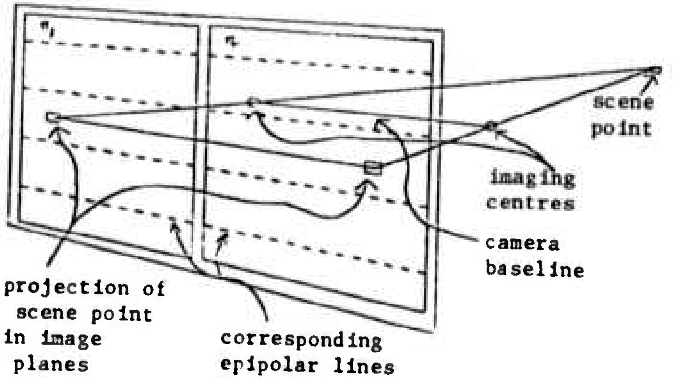

scene point
imaging centres
camera baseline
projection of scene point in image planes
corresponding epipolar lines

Figure 3-1: Collinear Epipolar Geometry from [Baker82]

We also give a bound on the disparity range allowable for any given segment, let us call it maxd.
Let $A=\{a_i\}$ be the set of segments in the left image
Let $B=\{b_j\}$ be the set of segments in the right image.
Then, for each segment a_i(resp. b_j) in the left (resp. right) image, we can define a <u>window</u> w(i)(resp. w(j)) in which corresponding segments from the right (resp. left) image must lie. The shape of this window is a parallelogram, one median being a_i(resp. b_j), the other a horizontal vector of length 2*maxd. One can see that a_i in w(j) implies b_j in w(i).
We define the boolean function p(i,j) relating two segments as:

- p(i,j) is true if
- b_j overlaps w(i)
- a_i , b_j have "similar" contrast
- a_i , b_j have "similar" orientation

The required similarity in orientation is loose and is a function of the segment length. We have set it to be 25 degrees for long segments and up to 90 degrees for very short segments.
Two segments are defined to have similar contrast if the absolute value of the difference of the individual contrasts is less than 20% of the larger one.
To each pair (i,j) such that p(i,j) is true we associate an <u>average disparity</u> d_{ij} which is the average of the disparity between the two segments a_i and b_j along the length of their overlap.
We define the two functions S1 and S2 as:

$$S1(a_i)=\{j|b_j \text{ in } w(i) \text{ and } p(i,j) \text{ is true}\}$$
$$S2(a_i)=\{j|b_j \text{ in } w(i) \text{ and } p(i,j) \text{ is false}\}$$

Similarly, we define $S1(b_j)$ and $S2(b_j)$. We will also need the value card(a_i), which is the number of elements in the set $S1(a_i)\ S2(a_i)$.
It is to be noted that all the functions described above are static, meaning that they are computed only once.

3.2. Description

Each possible match is evaluated by computing a measure of the distortion this match provokes for its neighbors,i.e. given that (i,j) is a correct match with its associated disparity d_{ij}, how well do the neighbors agree with this proposed disparity? We compute an evaluation of the match (i,j) and compare to the matches (i,k) and (h,j) for k in $S1(a_i)$ and h in $S1(b_j)$. If the evaluation is minimum for (i,j), then j is the preferred interpretation for i and i is the preferred interpretation for j. For any iteration after the first one, in order to evaluate a match (i,j), we only look at the preferred matches for the neighbors of i and j, if they have any. Formally, the computation of $v^t(i,j)$ is:

At iteration 1

$$v^1(i,j)=\left(\sum_{a_h\epsilon S_1(b_j)US_2(b_j)} \min_{\substack{b_k\epsilon S_1(a_h)\\ b_k\neq h_j}} |d_{hk}-d_{ij}|\right)\Big/card(b_j)$$
$$+\left(\sum_{b_k\epsilon S_1(a_i)US_2(a_i)} \min_{\substack{a_h\epsilon S_1(b_k)\\ a_h\neq a_i}} |d_{hk}-d_{ij}|\right)\Big/card(a_i)$$

At the end of each iteration, we define the sets $Q(a_i)$ and $Q(b_j)$ as

j in $Q(a_i)$ and i in $Q(b_j)$ if

$\forall k$ in $S1(a_i)$, $v^t(i,j)\leq v^t(i,k)$

AND

$\forall h$ in $S1(b_j)$, $v^t(i,j)\leq v^t(h,j)$

For any iteration after the first one, the computation of $v^t(i,j)$ becomes

$$v^t(i,j)=\left(\sum_{a_h\epsilon S_1(b_j)US_2(b_j)} \min_{\substack{b_k\epsilon Q(a_h)\\ b_k\neq b_j}} |d_{hk}-d_{ij}|\right)\Big/card(b_j)$$
$$+\left(\sum_{b_k\epsilon S_1(a_i)US_2(a_i)} \min_{\substack{a_h\epsilon Q(b_k)\\ a_h\neq a_i}} |d_{hk}-d_{ij}|\right)\Big/card(a_i)$$

if the sets Q are not empty, otherwise the computation of the function v is done using the formula for iteration 1.

At the last iteration, only those elements that have a preferred match are considered valid, and a disparity map array is filled using these values. It is interesting to note that this process is absolutely symmetric in the two views and therefore will yield identical results (except for the sign of the disparity) if the two views are interchanged. It is helpful to look at a simple example to understand this process.

3.3. Example

Let our 2 views be the ones shown in Figure 3-2 below:

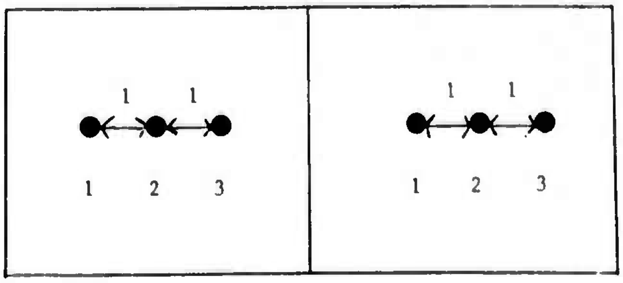


Figure 3-2: A simple example

In absence of any extra information, the correct interpretation is that the 3 points have the same disparity, and the result of the matching is (a_i, b_i) for i in $\{1,2,3\}$.
In this example, $S1(a_i)=S1(b_i)=\{1,2,3\}$ and $S2(a_i)=S2(b_j)=\emptyset$. The array d_{ij} is

$$\begin{matrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{matrix}$$

Therefore we find

$$\begin{aligned} v^1(1,1) &= (|d_{22}-d_{11}|+|d_{33}-d_{11}|)/3 \\ &+ (|d_{22}-d_{11}|+|d_{33}-d_{11}|)/3 \\ &= 0 \end{aligned}$$

compared to

$$\begin{aligned} v^1(1,2) &= (|d_{23}-d_{12}|+|d_{33}-d_{12}|)/3 \\ &+ (|d_{21}-d_{12}|+|d_{23}-d_{12}|)/3 \\ &= 1 \end{aligned}$$

and to

$$\begin{aligned} v^1(1,3) &= (|d_{22}-d_{13}|+|d_{32}-d_{13}|)/3 \\ &+ (|d_{12}-d_{13}|+|d_{11}-d_{13}|)/3 \\ &= 2.67 \end{aligned}$$

The calculations are similar for the other pairs, so, at the end of the first iteration, the preferred interpretations are only the correct ones, and further iterations will not alter the results.

3.4. Discussion

The criterion used here, namely the minimal differential disparity, has similarities with the edge interval constraints given in [Arnold80] and subsequently used by Baker[Baker 82], but looser in the sense that it does not require ordering of the edges. Since our criterion does not take ordering into account, a dynamic programming implementation is not possible. Our evaluation function is more informed than Baker's in the sense that it considers all edges in a neighborhood instead of just the predecessor and successor of a given edge. The performance of this algorithm on a few examples is presented next.

IV. Results

It is difficult to display results of stereo matching meaningfully, especially in a two dimensional picture, since we only generate a sparse disparity map. We will simply show the line segments in the two views that are found to match. We have not been able to master the art of cross-eyed stereo fusion, but since a number of people in the field are good at it, we will present all pairs of images according to its convention, that is the left view is shown on the right and the right view on the left. All results will also be shown this way, without explicitly marking each point and its correspondence. We first started our experiments with very simple line drawings, slightly more complex than the one shown in Figure 3-2 and the results matched the expectations. In order to remove the effects of the segmentation procedure on the performance of our matching technique, we hand-segmented the images shown in Figure 4-1 by tracing the boundaries of the objects on a digitizing table. This image, from Control Data Corporation, is synthetic and has been used by Baker[Baker82] for his experiments. The resulting segments are shown on Figure 4-2 and Figure 4-3 displays the results after matching. All the lines that have been matched have the correct correspondence, but some matches are missed. This is due to the fact that when the matcher gets confused by closely competing assignments, it chooses not to assign a label. Also, some edges are not matches because of mistakes in the tracing procedure: we traced the boundaries of some objects in opposite directions in the two views.
For all other examples, edge detection was performed automatically using a technique developed by Nevatia and Babu[Nevatia80] that finds edge magnitude and direction by convolving the image with edge masks in different orientations (we used 5x5 masks in 6 directions here). These edges are then linked to form boundary curves which are approximated by piecewise linear segments.

Next, consider the industrial part shown in Figure 4-4, the original resolution is 256 by 256 and the gray levels are coded on 8 bits. We applied the matching algorithm to two different resolutions of the image, running it through three iterations. It was found that no assignment was changed after three iterations in our experiments. Figure 4-5 shows the original edges and Figure 4-6 displays the results in the above mentioned form. Similarly, Figure 4-7 shows the segments at half resolution and Figure 4-8 the results. Looking at the segments one by one, we did not notice any spurious assignment at either resolution, meaning that we captured the shape of the object, even though the density of edges is much larger than in the previous example.

Another, more complex image is shown on Figure 4-9. In this image, we have a wide range of disparities, a change of sign in the disparities across the picture, various occlusions, the presence of a repetitive structure (a Rubik's cube) and contrast reversal. We do not expect to get good results with this contrast reversal since one of our preliminary conditions is similarity in contrast, but the other peculiarities are very interesting. We worked at low resolution on the segments shown in Figure 4-10 to obtain the results shown in Figure 4-11. The interesting points are the following:

- The elongated vertical blocks in the rear of the image are correctly put into correspondence.

- All the squares of the cube that should be identified are correctly matched. The correct labeling appeared at iteration 2 (at iteration 1, most of them are only ambiguously matched.)

The segments at high resolution are shown in Figure 4-12 and the matching results in Figure 4-13. We did not use the results at low resolution to guide the matching at high resolution, therefore the elongated block in the rear right is not matched any longer. It is interesting to note that the edges coming from the texture of the wood blocks do not create confusion, but help the matching, on the front cylinder for example. Once again, most assigned matches are correct.

V. Conclusions

This research is far from being in a final state. The initial encouraging results presented here must therefore only be viewed as an indication that the hypothesis of minimal differential disparity may be useful. The critical points that must be examined are:

- Relax the contrast constraint. This may be done by considering not the contrast of an edge, but the intensity values on each side. Edges could then be matched if either their left side or their right side correspond. One may eventually consider an edge as a doublet[Baker82] and match each side separately.
- To refine the formulation of the evaluation formula. Statistical analysis may yield better functions, maybe by introducing a static probability measure to evaluate each match based on similarity of intrinsic properties (length, color, orientation.) Also of concern is a more accurate definition of a no-match label, which is obtained if a match pair is not clearly better than the competing ones.
- Further extensive testing is also required on aerial and near range imagery, with terrain models for accuracy checking.
- Finally, we must use an interpolation scheme, very likely intensity-based, to generate a full disparity map of the scene depth.

VI. References

Arnold78 Arnold D. "Local Context in Matching Edges for Stereo Vision," in Proceedings of Image Understanding Workshop, Cambridge, Mass, May 1978, pp. 65-72.

Arnold80 Arnold D. and Binford T. "Geometric constraints in Stereo Vision," Society Photo-Optical Instr. Engineers, Vol. 238, Image Processing for Missile Guidance, 1980, pp. 281-292.

Baker82 Baker H. "Depth from Edge and Intensity Based Stereo," Stanford Artificial Intelligence Laboratory, AIM 347 Stanford, Calif., Sept. 82.

Barnard80 Barnard S. and Thompson W. "Disparity Analysis of Images," IEEE Trans. Pattern Anal. Machine Intell., PAMI-2,4 July 1980, pp. 333-340.

Barnard82 Barnard S. and Fishler M. "Computational Stereo," ACM Computing Surveys, Vol. 14, No. 4, Dec. 1982, pp. 553-572.

Binford81 MacVicar-Whelan P. and Binford T. "Line Finding with Subpixel Precision," in Proceedings of Image Understanding Workshop, Washington, D.C., Apr. 1981, pp. 26-31.

Gennery80 Gennery D. "Object Detection and Measurement Using Stereo Vision," in Proceedings of Image Understanding Workshop, College Park, Md., Apr. 1980, pp. 161-167.

Grimson79 Grimson W. and Marr D. "A Computer Implementation of a Theory of Human Stereo Vision," in Proceedings of Image Understanding Workshop, Palo Alto, Calif., Apr. 1979, pp. 41-47.

Grimson81 Grimson W. "From Images to Surfaces," MIT Press, Cambridge, Mass., 1981.

Hannah80 Hannah M. "Bootstrap Stereo," in Proceedings of Image Understanding Workshop, College Park, Md., Apr. 1980, pp. 201-208.

Henderson79 Henderson R., Miller R. and Grosch C. "Automatic Stereo Reconstruction of Man-Made Targets," SPIE, Vol. 186, No. 6, Digital Processing of Aerial Images, 1979, pp. 240-248.

Kelly77 Kelly R. ,McConnell P. and Mildenberger S. "The Gestalt Photomapping System," Journal of Photogrammetric Engineering and Remote Sensing, Vol. 43, No. 1407, 1977.

Lucas81 Lucas B. and Kanade T. "An Iterative Image Registration Technique with an Application to Stereo Vision," in Proceedings of Image Understanding Workshop, Washington, D.C., Apr. 1981, pp. 121-130.

Marr76 Marr D. and Poggio T. "Cooperative Computation of Stereo Disparity," Science 194, 1976, pp. 283-287.

Marr77 Marr D. and Poggio T. "A Theory of Human Stereo Vision," Memo. 451, Artificial Intelligence Laboratory, MIT, Cambridge, Mass., Nov. 1977.

Moravec80 Moravec H. "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," Stanford Artificial Intelligence Laboratory, AIM 340, Ph.D. Thesis, Sept. 1980.

Nevatia80 R. Nevatia and K. Babu, "Linear Feature Extraction and Description," Computer Graphics and Image Processing, Vol. 13, pp. 257-269, July 1980.

Panton78 Panton D. "A Flexible Approach to Digital Stereo Mapping," Journal of Photogrammetric Engineering and Remote Sensing, Vol. 44, No. 12, Dec. 1978, pp. 1499-1512.

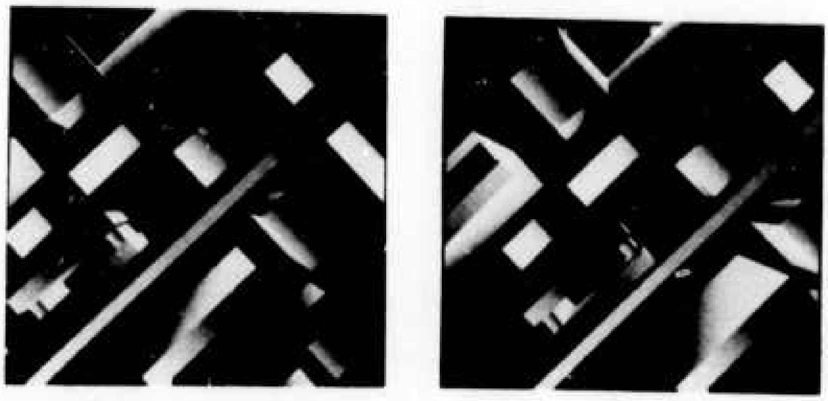


Figure 4-1: Synthetic image [256x256x6]

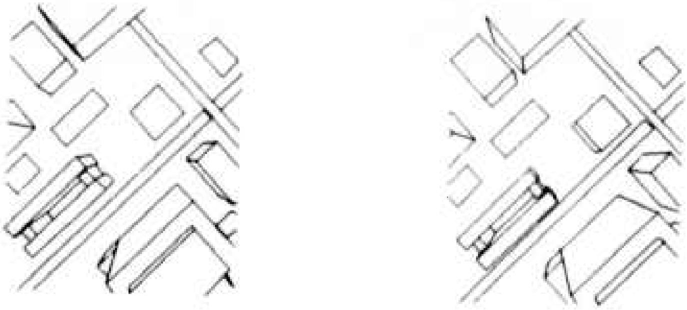


Figure 4-2: Hand generated segments

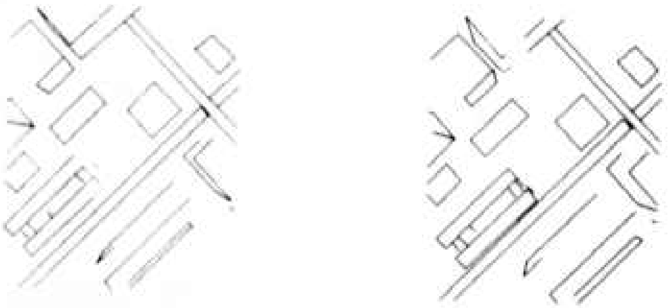


Figure 4-3: Results of the matching

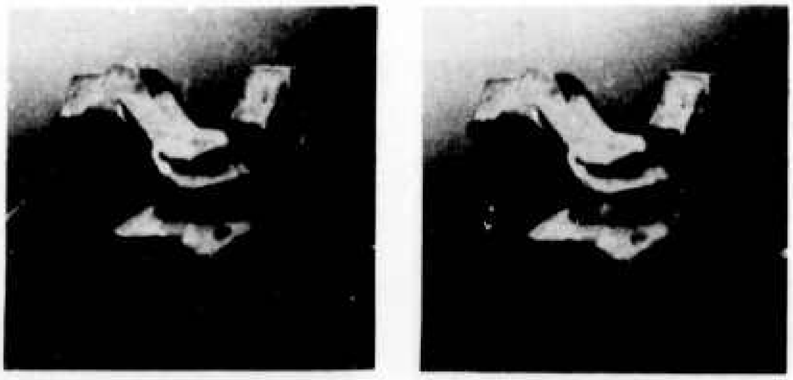


Figure 4-4: Industrial part [256x256x8]

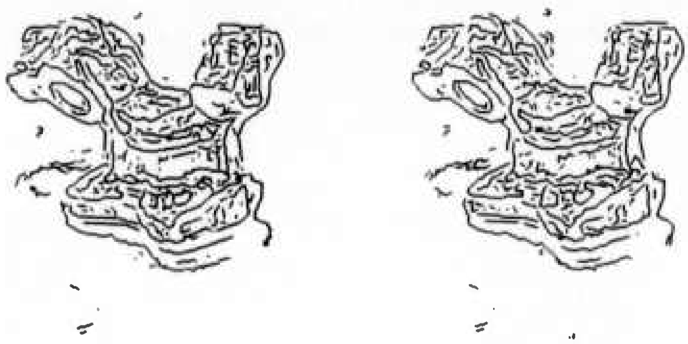


Figure 4-5: Segments from the full resolution image

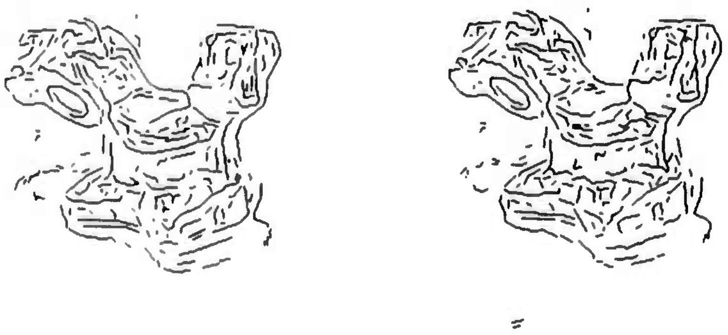


Figure 4-6: Results at full resolution

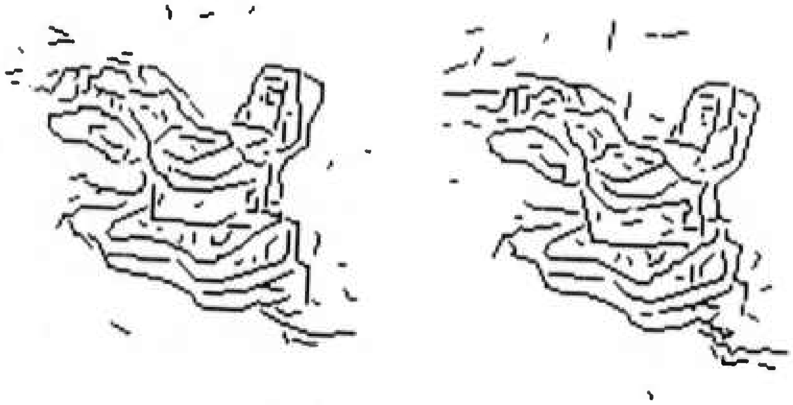


Figure 4-7: Segm~nts from the half resolution image

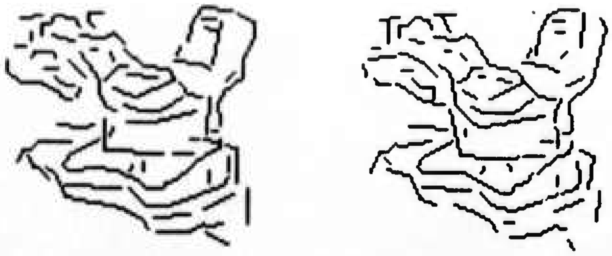


Figure 4-8: Results at half resolution

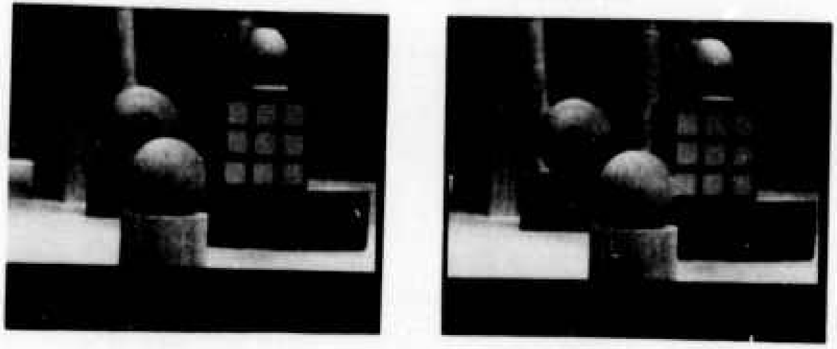


Figure 4-9: Image of some blocks[512x512x7]

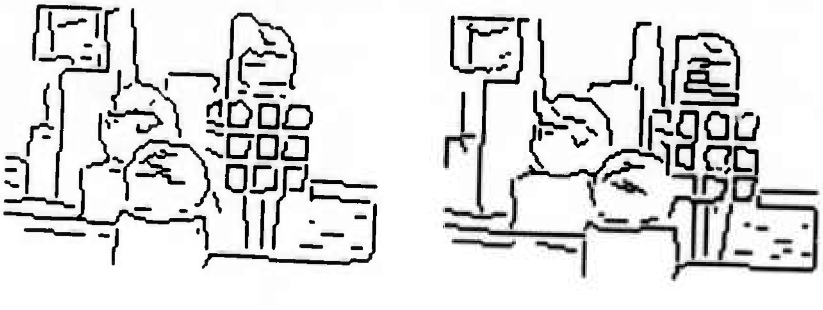


Figure 4-10: Segments at low resolution

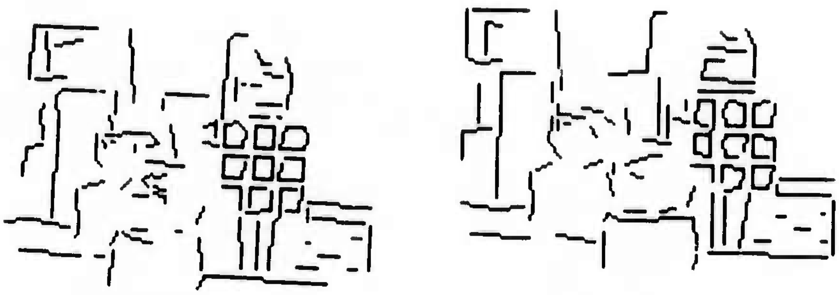


Figure 4-11: Results at low resolution

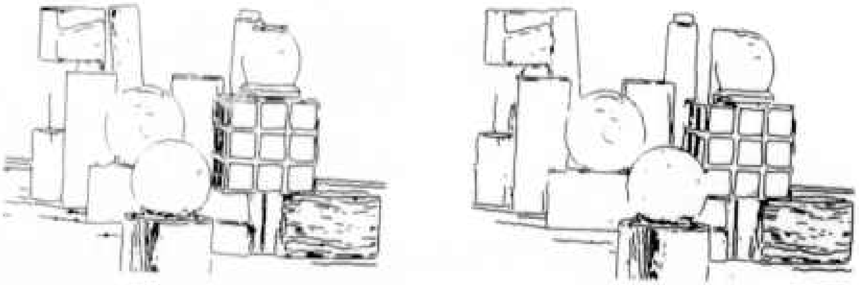


Figure 4-12: Segments at high resolution

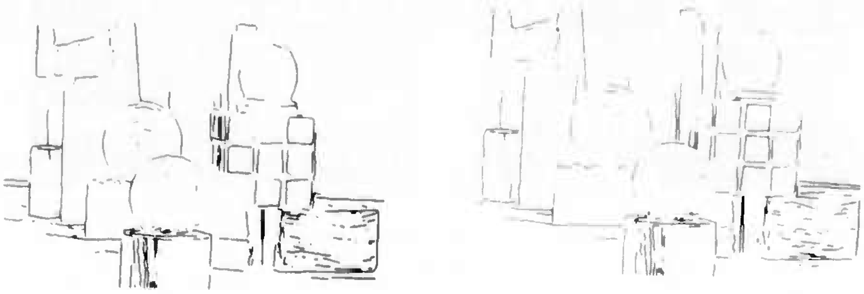


Figure 4-13: Results at high resolution

Software Metrics for Performance Analysis of Parallel Hardware*

R. David Etchells and Graham R. Nudd

Hughes Research Laboratories
3011 Malibu Canyon Road, Malibu, California 90265

ABSTRACT

The state of the art of parallel processing is characterized by an extraordinary proliferation of architectures. To date, little work has been done to quantify the relative performance capabilities of this range of architectures, especially from the viewpoint of general Image Understanding (IU) processing requirements. This paper discusses performance evaluation of parallel hardware. A set of software metrics is proposed, based on the processing requirements of common IU systems. The intent of the paper is to serve as a point of departure for further work and discussion.

OVERVIEW

The Image Understanding (IU) community is faced with an ever-expanding range of highly parallel computer architectures, many intended to uniquely match special processing requirements within that discipline.[1-6] A key issue in the development of such machines is the degree to which they meet the general needs of the field. To date, little attempt has been made to objectively analyse the performance characteristics of these machines, when applied to typical IU problems. The work that has been done in this area [7,8] has so far been limited in scope to just a few architectures, and to biomedical image processing application areas. No concerted effort has been made to study the full range of parallel architectures, within the broader context of Image Understanding and scene analysis. A partial reason for the lack of such comparative analysis is the absence of a set of widely accepted metrics for parallel hardware performance evaluation. Our purpose here is to propose a set of common algorithms for use as performance evaluation standards within the field of IU. Hopefully, the current paper will stimulate work leading to a set of software metrics that will be pertinent to IU, widely accepted, and simple to apply.

PERFORMANCE EVALUATION METHODS

In conventional numeric processing, there are two commonly used methods of performance evaluation. These are the instruction mix and benchmark program approaches. In the instruction mix approach, a set of programs is examined to determine the number of times each type of machine instruction occurs. The execution time of a particular processor performing the programs in question can then be estimated by multiplying its execution time for each machine instruction by the number of times that instruction occurred in the benchmark. The sum of all such products, one for each machine instruction, is then taken as the estimated execution time for the program set represented by the instruction mix.

The instruction mix approach is useful for rapidly evaluating the performance of conventional serial processors, but has little utility for the study of parallel architectures. This is because of the importance of data movement in IU applications. That is, two algorithms might show identical statistics, in terms of the numbers of multiplications, additions, etc. that each require, but show radically different execution times, due to widely differing data movement requirements. For example, one algorithm might involve only data taken from a relatively small kernel, while the other might require global access to data scattered across the entire

* This work is supported in part by the Defense Advanced Research Projects Agency of the Department of Defense, and was monitored by the Wright Patterson Air Force Base, under Contract F-33615-76-C-1203, DARPA Order No. 3119.

image data plane. Furthermore, the *pattern* of data movement is often at least as important as the amount of movement itself, since different architectures are able to take varying advantage of regularities in data movement. In addition to the importance of data movement, the instruction mix approach is unusable in IU applications because of varying efficiencies in parallel architectures. Many parallel processors, particularly SIMD arrays are not always able to use all of their available hardware to best advantage: Situations frequently arise in which a significant portion of the available hardware is idle, due to the lack of pertinent image data in the pixels associated with it. In such cases, a simplistic analysis based on aggregate instruction rates leads to performance figures significantly higher than can actually be attained.

The benchmark program method of performance evaluation, on the other hand, avoids many of the problems just mentioned. In this approach, a representative algorithm is programmed to run on a particular machine, and the actual execution time is measured. If the representative program is properly chosen, the results are virtually guaranteed to be accurate, since such matters as data movement, machine efficiency, and the operating environment are naturally included in the final measurement.

The problem with the benchmark program approach, of course, lies in the choice of the "representative program." Particularly in a field as broad as IU currently is, it would be a hopeless task to map *every* interesting algorithm onto *every* proposed architecture. The problem of selecting representative algorithms is itself complicated by the breadth of the field, the wide range of approaches to any given IU sub-task, and by the fact that there are many areas in which there is no clear consensus as to the best algorithm for performing a given task. These are the parameters within which we must work. They are further modified by a strong desire to reduce to an absolute minimum the amount of coding required to implement an evaluative test. Ideally, what we would like to do is to find the lowest level of program modules with the greatest degree of applicability across the entire range of IU algorithms.

In selecting representative algorithms or modules though, we must be particularly careful to not only represent the full range of application requirements (such as feature extraction, classification, etc.), but to include as well the entire range of processing requirements, as seen by the hardware. In other words, while covering the entire range of IU algorithms, we must (since our objective is *hardware* evaluation) focus more on the processing load in making our selections, rather than on the overall structure of any particular algorithm. Accordingly, we need to develop a conceptual basis, or taxonomy, by which we might determine the unique processing requirements of each algorithm studied.

SOFTWARE TAXONOMIES

Little work has been done to date on the classification of software algorithms. Swain, Siegal, and El-Achkar [9] proposed a six-point classification scheme which consisted of the following catagories:

● Type:	- Enhancement - Extraction
● Context Dependency:	- Context Free - Context Dependent
● Iteration:	- Single-Pass - Multi-Pass
● Multivariacy:	- Univariate Data - Multivariate Data
● Time:	- Real-Time - Batch
● Computational Complexity:	- n, n log(n), etc.

Swain's "type", "iteration", and "time" classifications are self-explanatory. An example of a "context-free" algorithm might be histogramming, where the set of final values is solely a function of the values of the individual pixels, independent of any relative associations that might exist between pixels. An example of a "context-dependent" algorithm would be one performing adaptive filtering in which the output value (and, indeed, the structure of the algorithm itself) for a given pixel would depend strongly on the values of the pixels surrounding it. A simple grey-scale image would fall under the catagory of "univariate data", while a Landsat multi-spectral image would be considered "multivariate data". Finally, "computational complexity" refers to the relative dependence of the execution time of an algorithm on the size (n) of the data being manipulated.

These classification criteria focus more on the *use* to which various algorithms are put than on their structure. As a result, little explicit information is conveyed regarding the processing requirements of the algorithms so classified. We propose as a more useful set of criteria the following:

- Functional Statistics
- Local vs. Global
- Memory Intensive vs. Computation Intensive
- Context Dependent vs. Context Free
- Iconic vs. Symbolic
- Object Oriented vs. Coordinate Oriented

Here, the term "functional statistics" simply refers to statistics of the sort normally used in compiling representative instruction mixes. In particular, we refer to the relative frequency of various arithmetic operations, such as addition, multiplication, division, etc. Such statistics are important in evaluating the performance characteristics of specific machines, but are less valuable in the study of general architectures. Apart from gains attributable to the level of parallelism employed, arithmetic performance is more a function of implementation than architecture.

The local/global distinction in our classification scheme is really a measure of the a priori knowledge concerning data location contained in an algorithm, rather than an indication of the size of the domain upon which an algorithm operates. That is, we consider the scope of an algorithm to be "global" if its domain cannot be, a priori, restricted to any subset of the image data. Thus, a "global" algorithm is one which may draw its data from anywhere in the image plane, whether or not it does so in all cases. This distinction arises from the scope of the data access required by the individual processing elements in a parallel architecture. If an algorithm may require a processor to have access to any area of the image, the architecture must provide for such arbitrary access. From the viewpoint of the computer architect, the fact that only a small number of pixels will be involved in a given operation matters less than the fact that those pixels may lie anywhere in the image plane.

The memory/computation intensive classification is a measure of the amount of local memory that will be needed to successfully execute an algorithm. "Local memory" is taken to mean that memory associated with each sub-processor in a parallel machine. This memory is used to store such things as raw image data, convolution coefficients, or intermediate results. An example of such usage would be the need to store edge magnitudes for each of several edge directions, in most edge-detection algorithms. Operations based on sorting within a kernel (eg: median filtering) in particular require large amounts of local memory.

As with Swain's classification, we take context dependency to mean the extent to which the values output by an algorithm depend on relationships existing between input data elements. Note that this definition of context dependency does not refer to situations in which the output of an algorithm depends non-linearly on the input values (as with thresholding). Such behavior is described by our linear/non-linear classification. The more commonly employed term of "data dependency" refers to both context dependency and linearity. We have chosen to distinguish these two cases as separate classification parameters because of their different implications for hardware.

Our "iconic vs. symbolic" catagorization is included because the two representations involve greatly different types of processing. In iconic processing, there is a direct relationship between physical storage locations and image pixels. Symbolic processing, on the other hand, involves the manipulation of lists and other data structures which contain image coordinates only as explicit entries in the data structure. Machines well suited to iconic processing are largely unsuited to symbolic processing, and vice versa. Most of the concurrent architectures proposed to date have been of the iconic type.

The problem of iconic vs. symbolic processing goes beyond the simple dichotomy of the classification, however. Modern image understanding frequently involves the translation of image data from the initial, iconic, form to a subsequent, symbolic one. A great deal of the processing load of advanced, autonomous systems actually occurs on the symbolic level, in the application of knowledge-based rule systems to the raw data gathered by the lower levels of the vision system. Both sorts of processing are therefore important to practical applications. Significantly, though, while we know fairly well how to build machines that are capable of processing either iconic or symbolic data, no current architectures adequately address the problem of translation between the two domains. The difficulty of this translation lies in the fact that it is basically an object-oriented process. The data corresponding to a particular object might lie anywhere within the image plane, making it difficult to make any a priori assignments of individual processors to individual objects in a multiprocessor architecture. Similarly, SIMD machines can only translate between iconic and symbolic representations one object at a time, due to their single instruction stream. There are other considerations involved here that extend beyond the scope of this paper, but suffice it to say that the iconic/symbolic translation problem remains difficult and as yet unsolved. It is for this reason that we have included "object orientated vs. coordinate oriented" in our list of classification parameters. Coordinate oriented processing refers to situations in which the location of the data to be processed within the image is known in advance, independent of any characteristics of that data. On the other hand, in object-oriented processing, the location of the data to be processed is an implicit function of the data itself, and of relationships existing within the data. The consequence for hardware, as mentioned earlier, is that object-oriented processing requires access to the entire image plane. Few architectures provide such access while allowing independent processing of various parts of the image.

This list of classification catagories provides a basis for a study of algorithm characteristics, as distinguished by the demands placed on the processing hardware. We will use them subsequently in our discussion of IU processing requirements, and again in our overview of current architectures.

IU ALGORITHM OVERVIEW & METRIC SET

In this section, we briefly review the most common types of processing encountered in image understanding. Before launching directly into this discussion, though, it would be appropriate to consider the level of algorithms that would be most profitable to study. We would like to find algorithms or operations which enjoy wide application across the entire IU field. We must balance this desire against the requirement that the algorithms selected be uniquely representative of the requirements of IU, as identified by our taxonomy. Obviously, the operations of addition and multiplication are widely used within IU. By themselves, though, they do little to represent the unique requirements of the discipline. On the other hand, the "Smith, Smith, Smith, and Jones" matched filter for '57 Chevys, while highly developed, has only a limited range of application.

Accordingly, we have selected a set of "unit operations" which function at a low enough level that they may be employed by a wide range of higher level algorithms, but that are themselves of a sufficiently high level to be classified according to our previously developed taxonomy. Table I lists the unit operations that we have selected for consideration, and shows how they fit into our classification scheme. A discussion of the unit operations and their classification follows.

It is important to note in the following discussion that many of the operations described can be used in ways contradictory to their primary classification. This does not invalidate in any way their selection as part of the metric set, based on our classification of them. Our intent here is not to rigorously classify the algorithms, including all variations of their usage, but merely to insure that we have adequately accounted for the various types of processing represented by our taxonomy.

Thresholding, the first entry in Table I, finds broad application throughout IU. It was chosen for inclusion in the metric set as the simplest example of a parallel, non-linear operation. As for its other parameters in the classification scheme, it is local, because thresholding by definition takes as input only the values of individual pixels. Thresholding may be either context-free or context-dependent, depending on whether it is being done adaptively or not. If the threshold value is the same for all pixels of the image, the operation is context-free. On the other hand, if the threshold is set locally, as some function of local data values, the operation is context-dependent. Examples of both types of thresholding would be good candidates for inclusion in the metric set. Since thresholding does not typically involve the storage of intermediate results, little local memory is required, and the operation is therefore considered to be computation-intensive. Thresholding is a coordinate-oriented operation, even in the adaptive case, because the data required to generate a given result always lies within a small area surrounding the pixel being processed. Likewise, the operation is strictly iconic.

Convolution, the second table entry, is widely employed in filtering functions such as edge detection, and as part of such procedures as connectivity linking, and region growing. It is basically a linear, arithmetic process, in which the data values within a local neighborhood are multiplied by a set of weight values, and the resulting products are summed to produce the final result. Such a sum-of-products is computed for each pixel of the input image. As we have just stated, convolution is an example of a linear, local operation. In some situations, the output is made non-linear, but this occurs through thresholding, which has already been included in the metric set. In its purest form, convolution is context-free, with the weighting function being invariant across the image. In some forms of adaptive filtering, a multi-pass iteration is applied, with the local weighting functions being modified by the results of the earlier pass. An example of such usage would be an algorithm to extract the lines forming the loops and whorls of fingerprints. In this application, the weighting values of a line-detecting filter are modified according to the dominant local line direction found on a previous pass. As with thresholding, both adaptive and non-adaptive forms of convolution processing should be included in the metric set. Convolution by itself is computation- rather than memory-intensive. Some of its applications do involve the storage of intermediate products, however. Edge-detection, for example, usually involves a series of convolutions, one in each edge direction being tested for, with the intermediate results of each individual convolution being stored for subsequent comparison and selection of the largest directional value at each point. We consider this to be an example of sorting, though, which we treat separately as the third entry of the table. Adaptive filtering can also involve substantial amounts of local memory, depending on the architecture of the machine being used. In some machines, particularly cellular arrays, the various weighting coefficients for each of a range of possible convolutions are all stored in the machine's local memory. This is the primary incentive for including adaptive convolution in the metric set. As to its other classification parameters, convolution is strictly coordinate oriented and operates within an iconic representation.

Sorting, the third entry in Table I, is included as an example of a local, non-linear, memory-intensive process. As just mentioned, it finds application in operations such as line-finding, where the largest of a set of several values must be selected. Median filtering likewise involves the selection of the median value from among a number of data values occurring in a local neighborhood. (Median filtering is most often used for size discrimination and connectivity processing.) Sorting is best thought of as context-dependent, since the shuffling of pieces of data or the setting of pointers is strictly a function of the data values themselves. It is also coordinate-oriented, but can occur in either an iconic or symbolic representation.

Histogramming is our fourth candidate for inclusion in a set of software metrics. Histogramming is a representative of what might be called "statistical processing," and constitutes a large portion of the computational load of the popular region-splitting segmentation algorithms. Its processing requirements differ from those of the operations detailed so far, in that it operates globally in a context-free fashion. It is most properly thought of as computation-intensive, since memory is only required for the storage of the final tallies. On the other hand, on MIMD and pipelined machines, its actual execution is memory access-intensive, in that a small set of memory locations are accessed repeatedly as the individual tallies are updated. Histogram computation is also linear with respect to the input values, and is most often employed in an iconic context. While we have classified histogram processing as being coordinate-oriented, it could be argued that many applications use it in a object-oriented manner. An example of such usage would be situations in which histograms are generated for a class of objects within the image, as is common in target-identification routines. This object dependency is not an intrinsic characteristic of the histogramming process, though, but rather an extra, externally-applied condition. Hence our "coordinate-oriented" classification.

Correlation operations were selected as the fifth metric set candidate because they involve local processing with a higher level of context-dependency than we have previously encountered in this discussion. As typically applied (in stereo processing), portions of one picture are compared against various regions of another picture. The comparison process is basically a convolution, but the weighting functions are the data values of the reference image. Correlation thus tests an architecture's ability to rapidly access different sets of weighting values for convolution processing. This sort of operation is common in both feature- and intensity-based stereo processing. In most implementations, it is also somewhat object-oriented, in that the correlations are performed only in areas containing features of interest. These areas of interest may lie anywhere in the image plane, and so could be thought of as fitting the description of "object-oriented." On the other hand, this "object orientation" is usually an attempt to reduce the computational load on conventional serial computers. The intrinsic structure of the processing implies no object orientation, and so we classify it as coordinate-oriented. Correlation is also a linear process, and usually is performed in the iconic domain.

Interior point selection, our sixth candidate for the metric set, is the process of identifying those points of the image that lie within closed boundaries defined by previously located edge segments. A point is determined to be on the interior of a closed boundary if it there are edge segments within a certain radius of it, in a majority of the directions checked. Since a point is either inside an object or not, the processing is non-linear, and since the data may lay anywhere within the image plane, the processing is global. It is furthermore both context-dependent and object-oriented, according to the definitions given earlier. Finally, it most naturally operates on iconically represented data, and is computation intensive, in that little intermediate data is stored for each point evaluated.

Line finding is the seventh metric candidate we have considered. By "line finding," we mean those routines which are concerned with linking edge segments together into lines, and "tracing" the resulting lines to determine their lengths and orientations. This is the most clearly object-oriented process that we have considered so far, in that the "tracing" operation necessarily involves following the line wherever on the image plane that it might go. The process is particularly interesting, because it operates on iconically represented data to produce symbolic data. As mentioned earlier, and as we shall see subsequently, in the discussion of machine architecture characteristics, such translation processes pose particularly difficult problems to computer architects. On array machines, the process is computation-intensive according to our earlier definition, because the generated line data remains distributed across the memories of a number of the cellular processors. On the other hand, on MIMD machines (which are generally better suited to this sort of processing), the process is memory intensive, requiring large amounts of local memory for its efficient execution. As to other classification catagories, the operation is non-linear, global, and context-dependent.

Shape descriptions are the eighth member of our metric set. Shape description, like line-finding, involves translating information from an iconic to a symbolic representation. As such, it is a strongly object-oriented process, involving step-by-step tracing of boundaries, or repeated computation of individual line-segment midpoints. Representative algorithms in this catagory include invariant moment calculations, medial axis transforms, and generalized cones. Shape description algorithms are usually global, context-dependent, and computation-intensive, according to our classification scheme. They are also linear, in that their output depends linearly on the shape characteristics of the input structure.

Our two final entries in the proposed metric set are examples of more purely symbolic processing. Graph matching, the first of our symbolic metrics, involves searching a graph for a sub-graph having a particular, specified structure. Prediction, the final metric candidate, involves the application of rules to a set of existing data to predict the probablility of occurrence of some particular condition.

Symbolic processing does not fit easily into our classification scheme, in that virtually all such computation is described similarly within the taxonomy. An extension to the classification method is doubtless in order. A paper[10] by Hillis indicates four possible catagories which might be used to describe symbolic processing. Hillis' list of critical operations for symbolic computation may be paraphrased as follows:

- **Deduction** of facts from semantic inheritance networks.
- **Matching** of patterns against sets of assertions, demons, or productions. Best matches must be selected in the absence of a perfect match.
- **Sorting** of sets according to chosen parameters.
- **Searching** graphs for sub-graphs with a specified structure.

Our proposed graph matching metric directly addresses the fourth of Hillis' catagories, while our prediction metric is more generally directed at the full range of processes.

HARDWARE ANALYSIS

Having established a basis for understanding the processing requirements of IU, we now turn to an evaluation of parallel hardware. Here, we examine several representative classes of architectures, from the standpoint of their abilities to perform the various types of processing described by our software taxonomy. Table II is a matrix showing the relative ability of each of these architectures to perform each of the classes of processing discussed earlier. In the matrix, processors are evaluated on a five-point scale, ranging from "++", for a machine that is highly suited to the type of processing represented by that column of the matrix, to "--" for an architecture that is highly unsuited to that type of processing.

The table lists eight processor catagories: cellular numeric, pipelined, MIMD, number theoretic, systolic, "broadcast", data-driven, and associative. There can be some overlap between these catagories, but the principle characteristics of each class are sufficiently distinct to permit discussion. In the following, we shall describe each architecture briefly, and examine how well it meets the processing requirements represented by our previously-developed software taxonomy.

Cellular numeric machines are those composed of an array of identical processors, or cells, directed by a common instruction stream, but operating on separate data. The processor array is usually the same size as the input image data array, with one processing cell assigned to each pixel of the input image. Cellular machines are perhaps the most popular of all the classes listed, with many already built or planned.[11-15] Their hardware advantages are great parallelism, high regularity, and simple circuitry in each of the cells. These attributes combine to make for relatively simple design and VLSI layout. Cellular machines typically employ nearest-neighbor communication between array members, although some provision is usually made for rapidly propagating or "broadcasting" data values across the array as a whole. Local memory for each processing cell is usually fairly limited.

From a software standpoint, because of their great parallelism, cellular machines are excellent for local, linear processing, such as convolution. They are somewhat less efficient at non-linear or context-dependent processing due to their single instruction stream. Cellular arrays usually execute context-dependent algorithms through an exhaustion process, generating all possible results, and then selecting the most appropriate for each pixel through a thresholding operation. Due to their limited local memory capacity, they are also less well suited to memory-intensive algorithms such as sorting, which requires the storage of a number of pieces of data at each array location. Their next-neighbor communication scheme can also be limiting in algorithms requiring a great deal of data sharing across a large area. A global-broadcast capability does permit excellent performance at such tasks as histogramming, where a large number of values must be compared with all data in the array. Most cellular machines to date are structured for iconic data processing, and are rather unsuited for symbolic processing. Cellular machines also have difficulty performing object-oriented tasks. Due to their single instruction stream limitation, they waste most of their parallelism in such cases, having to deal with only a single object at a time.

Pipelined machines are ones in which data is operated on by a chain, or "pipe" of hardware units, each performing a separate function. As each element of the chain finishes its operation, it passes its results to the next in line, and accepts a new piece of data from the previous unit. Processing occurs simultaneously in all elements of the chain, with data flowing down the chain as liquid down a pipe. Concurrency is obtained through having a number of operations take place simultaneously along the pipe, and sometimes by having a number of pipes operating simultaneously. Pipelining is fairly common in commercial "array processors." Several research machines have been using this architecture for use in IU.[16,17]

Since they rely upon fixed <u>sequences</u> of operations within algorithms, pipelines exhibit high performance in situations in which the type and sequence of operations to be performed is rigidly determined. This is the case in many lower-level algorithms, such as thresholding and convolution. The penalty paid for the speed thus obtained is that there is a fixed delay from the time data is first presented at the beginning of the "pipe," and the time when results are first available at the end.

This delay corresponds to the amount of time it takes for the data to pass through all the functional units along the pipe. As a consequence, a heavy time penalty is usually paid for any data-dependent program branching. This is because, each time a branch is taken, the pipe must be "filled" with new data before the new results become available. For this reason, pipelines perform less well on nonlinear, context dependent, and object-oriented algorithms. Machines with multiple pipes can also be limited by memory contention problems, if the pipes share a common memory. (Hence the "average" rating for pipelines under the "global" and "coordinate-oriented" headings in Table II.) Pipelines are also unsuited for symbolic processing, due to the high degree of context-dependency involved.

MIMD stands for "Multiple Instruction, Multiple Data," and refers to machines in which a number of largely independent processors are harnessed to process data in parallel. Each of the processors in an MIMD architecture execute their own instruction stream. A significant factor motivating the development of such machines is the wide availability of cheap, general-purpose microcomputer chips. (Computer architects are irresistably led to ponder the power of a thousand, or better yet, a million Z-80 chips working in parallel.) Several such machines have been built for use in IU, and are presently being studied.[18,19]

Since the individual processors in MIMD machines are usually general-purpose ones, with large memory spaces and local program storage, the architectures as a class handle context-dependent and memory-intensive processing with greater ease than do cellular and pipelined systems. They also perform object-oriented processing better than either of these architectures, due to their large numbers of relatively autonomous processors. The principle drawbacks of MIMD machines are the difficulties of coordinating the operation of so many asynchronous processors (interprocessor communications), and the problem of partitioning the image data across the available processor memories. Communication problems arise when processors must access data from other processors' memories, and this accounts for the "average" ratings for MIMD machines in the "global," "object-oriented," and "coordinate-oriented" catagories in Table II.

Number theoretic and systolic processors are two special catagories of machines particularly suited to linear, context-free, computation-intensive processing. They are thus highly suited to the lowest and most computation-intensive levels of image understanding, but have little application for higher levels involving context-dependent, object-oriented, or symbolic processing. Number theoretic processors employ the residue number system, studied extensively by Szabo and Tanaka,[20] in which integers are represented by their "residues" with respect to each of a set of relatively prime numbers called "moduli." A residue of x mod m, is the least positive integer remainder of the division of x by m, where x is the number to be converted, and m is one of the moduli. If M is the product of the moduli used in such a system, integers in the range of $1 \leq x \leq (M-1)$ can be uniquely represented by their sets of residues. The utility of the residue system lies in the manner in which arithmetic operations are performed. Essentially, all operations are done in parallel, mod m_i, for each of the moduli. After all required operations have been performed, the residue representation of the result is converted back into a binary form. The fact that the processing in residue form is all done modulo m_i, where the m_i are the moduli means that multiplications can be reduced to table-lookups, without requiring prohibitively large ROMs. Consequently, computation-intensive tasks can be performed extremely rapidly. A processor for operating on 5 x 5 kernels has been successfully built and tested using this approach.[21] The drawbacks to the technique are that operations such as thresholding are difficult to perform in the residue domain, and the overhead from converting from binary to residue formats and back again makes the technique less advantageous for situations in which relatively little processing is to be done in residue form.

Systolic processors are a class of array machines exhibiting extreme regularity of hardware. Their name derives from the fact that processing within the array occurs in the form of "waves," moving from one edge to the other. Each processing element in the array takes data in from some of its neighbors on one cycle, processes it, and passes the results on to its other neighbors a cycle or two later. The high regularity of the hardware and data movement within such arrays makes them ideal candidates for VLSI implementation, and the high degree of parallelism attainable results in very high processing throughputs. Unfortunately, they are rather weak in the areas of context-dependent, object-oriented, memory-intensive, and symbolic processing, and so are largely unsuited for the higher levels of IU processing. Several machines have been built, however, and have demonstrated excellent performance in the domains of their greatest utility.[22,23]

We call "broadcast" machines those in which individual processors communicate with each other through some sort of "message net," in which data paths are established through the inclusion of routing tags appended to the data being transmitted. Architectures of this sort have been primarily developed for symbolic processing. The overhead associated with their communication method makes them largely unsuited for computation-intensive iconic processing, in which the flexibility of the broadcast communication scheme is not required. Some variation of the communication technique may prove useful for the difficult task of object-oriented iconic to symbolic translation. A machine using a "broadcast" architecture is currently being constructed at MIT,[10] for processing semantic nets.

Data-driven processors are machines composed of a number of execution units, each of which performs some simple, low-level function. These units are interconnected according to requirements of the program being implemented. That is, if for instance, a multiplication involves the results of two earlier addition, the outputs of two addition execution units would be connected to twc inputs of a multiplication unit. The execution units function independently and asynchronously. Whenever a unit has all of the operands it needs to generate a result, it "fires," and passes its results to any other units to which it might be connected. Dennis and Misunas[24] have done much to popularize data-flow architectures (the more popular term describing this class).

Data-driven machines promise to achieve very high throughputs for computation-intensive numerical processing, because they eliminate the instruction bottleneck common to so many other architectures. Their limitations for use in IU, though, include limited amounts of intermediate memory, and reduced performance for highly context-dependent memory. This last arises from the overhead associated with changing the configuration of the processing elements in response to rapidly changing program requirements. A means of avoiding this restriction has been implemented[25] by expressing the execution units in the software of an MIMD machine. In this variation, the "execution units" are instructed cycle by cycle what operations they are to perform. This greatly increases the flexibility of the resulting machine, but creates a new bottleneck in the hardware responsible for task assignment.

Associative processors, our last architecture catagory, are really more of an attribute than a separate class of architectures. Many of the architectural classes discussed earlier, such as cellular numeric or MIMD may be given an associative capability simply through the addition of the appropriate hardware. Architectures incorporating associative capability usually gain their associative power at the expense of numeric processing capacity. This accounts for the low or "average" ratings of associative machines for most catagories in Table II. The strongest point in favor of associative machines (in the present context, at least) is the exceptional symbolic processing performance of which they are capable. They derive this performance from the fact that they may search sets or graphs for particular elements or nodes in parallel, for all members of the structure being scanned. STARAN[26] is probably the best-known associative machine constructed to date.

CONCLUSIONS

In the foregoing, we have identified six sets of software characteristics which have particular relevance to the type of processing demanded by the algorithms so described. We have measured the capabilities of a range of machine architectures against the processing requirements implied by each of these software characteristics, and have tabulated the results. Perhaps the most valid conclusion to be drawn from this analysis is that no single architecture is capable of performing all classes of processing equally well. More significant though, is the fact that, while machines exist that are well suited for either iconic or symbolic processing, no present architecture is efficient at the task of translating iconically represented data into a symbolic representation. This is of key importance for future real-time knowledge-based IU systems, in that they must be able to rapidly and effectively process iconic data, and subsequently input that data to the knowledge-based portions of their structure. The problem to be solved is one of concurrently processing object-oriented data, without running afoul of communication or memory-access bottlenecks. The form of the ultimate solution to this problem is the subject of active investigation by many researchers. With the present development in the low-level numeric architectures, and the current activity in symbolic processors, this topic is an area of fruitful research.[27]

REFERENCES

[1] Proc. 1981 IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management, Hot Springs, Virginia, Nov 11-13, 1981, IEEE Press, New York, 1981.

[2] Fu, K.S. and Ichikawa, T., Special Computer Architectures for Pattern Processing, CRC Press, Florida, 1982.

[3] Duff, M.J.B., and Levialdi, S., Languages and Architectures for Image Processing, Academic Press, London, 1981.

[4] H.T. Kung, ed., VLSI Systems and Computations, Computer Science Press, 1981.

[5] Proceedings of the Fifth International Conference on Pattern Recognition, Dec 1-4, 1980, Miami Beach, IEEE Computer Society Press, New York, 1981.

[6] Proceedings of the Sixth International Conference on Pattern Recognition, Oct 19-22, 1982, Munich, Germany, IEEE Computer Society Press, New York, 1982.

[7] Preston Jr., K., "Comparison of Parallel Processing Machines: A Proposal;" in [3], pp.305-319.

[8] Preston Jr., K., "Image Manipulative Languages: a Preliminary Survey;" in Pattern Recognition in Practice, Kanal, L.N. and Gelsema, E.S. (eds.), North-Holland, Amsterdam, 1980.

[9] Swain, P.H., Siegal, H.J., and El-Achkar, J., "Multiprocessor Implementation of Image Pattern Recognition: A General Approach," in [5], pp. 309-317.

[10] Hillis, W.D., "The Connection Machine (Computer Architecture for the New Wave);" MIT A.I. Memo #646, September, 1981.

[11] Etchells, R.D., Grinberg, J., and Nudd, G.R., "The Development of a Three-Dimensional Circuit Integraton Technology and Computer Architecture;" Society of Photographic and Instrumentation Engineers, Vol. 282, pp. 64-72, Washington D.C., April, 1981.

[12] Reeves, A.P., "A Systematically Designed Binary Array Processor;" Trans. IEEE on Comp., C-29, pp. 278-287, 1980.

[13] Batcher, K.E. "Design of a Massively Parallel Processor;" Trans. IEEE on Comp., C-29, pp. 836-840, 1980.

[14] Duff, M.J.B., "Review of the Clip Image Processing System;" Proceedings of National Computer Conference, pp.1055-1060, 1978.

[15] Hunt, D.J., "The ICL DAP and its Application to Image Processing;" in [3], pp.275-282.

[16] Gerritsen, F.A. and Aardema, L.G., "Design and Use of DIP-1: a Fast, Flexible and Dynamically Microprogrammable Pipelined Image Processor;", Pattern Recognition, 14, pp. 319-330.

[17] Granlund, G.H., "The GOP Parallel Image Processor;" in Digital Image Processing Systems (eds. Bolc and Kulpa), Springer-Verlag, Berlin, pp. 201-227, 1981.

[18] Kruse, B., Danielsson, P.E., and Gudmundsson, B., "From PICAP I to PICAP II;" in [2], pp. 127-156.

[19] Rieger, C., "ZMOB: Doing it in Parallel;" in [1], pp. 119-124.

[20] Szabo, N., and Tanaka, R., Residue Arithmetic and its Applications to Computer Technology, McGraw-Hill, New York, NY, 1967.

[21] Fouse, S.D., Nudd, G.R., and Cumming, A.D., "A VLSI Architecture for Pattern Recognition Using Residue Arithmetic"

[22] Nash, J.G., Hansen, S., and Nudd, G.R., "VLSI Processor Arays for Matrix Manipulation;" in VLSI Systems and Computations, pp. 367-378, ed. by H.T. Kung, Computer Science Press, 1981.

[23] Blackmer, J., Frank, G., and Kuekes, P., "A 200 Million Operations per Second (MOPS) Systolic Processor;" Proceedings of SPIE Real-Time Signal Processing IV, pp. 10-18, August 25-28, 1981, San Diego.

[24] Dennis, J.B., and Misunas, D.P., "A Computer Architecture for Highly Parallel Signal Processig;" Proceedings of the ACM 1974 National Conference, ACM, New York, November 1974, pp. 402-409.

[25] Guzman, A., "A Parallel Heterarchical Machine for High-Level Language Processing;", in [3], pp. 229-244.

[26] Batcher, K.E., "STARAN Series E;" Parallel Processing Symposium, August 1977.

[27] Etchells, R.D., and Nudd, G.R., "VLSI Image Understanding Systems;" Technical Report: Image Understanding Research, University of Southern California IPI, October 1983. Ed. by R. Nevatia. (Forthcoming)

12977-1

OPERATION	LOCAL/ GLOBAL	LINEAR/ NON-LINEAR	MEMORY INTENSIVE/ COMPUTATION INTENSIVE	OBJECT ORIENTED COORDINATE ORIENTED	CONTEXT FREE/ CONTEXT DEPENDENT	ICONIC/ SYMBOLIC
THRESHOLDING	L	NL	CI	CO	EITHER	I
CONVOLUTION	L	L	CI	CO	EITHER	I
SORTING	L	NL	MI	CO	CD	I
HISTOGRAMMING	G	L	CI	CO	CF	I
CORRELATION	L	L	CI	CO	CD	I
INTERIOR POINT SELECTION	G	NL	CI	OO	CD	I
LINE-FINDING	G	NL	EITHER	OO	CD	TRANSLATION
SHAPE DESCRIPTIONS	G	L	CI	OO	CD	TRANSLATION
GRAPH MATCHING	–	NL	MI	OO	CD	S
PREDICTIONS	–	NL	MI	OO	CD	S

Table I. Software Metric Characteristics

12977-2

ARCHITECTURE	OPERATION CLASS												
	LOCAL	GLOBAL	LINEAR	NON-LINEAR	CONTEXT-FREE	CONTEXT-DEPENDENT	MEMORY INTENSIVE	COMP INTENSIVE	OBJECT ORIENTED	COORDINATE ORIENTED	ICONIC	SYMBOLIC	TRANSLATION
CELLULAR NUMERIC	++	+	++	0	+	0	–	+	– –	+	+	– –	– –
PIPELINED	++	0	+	0	+	0	–	+	–	0	+	– –	–
MIMD	++	0	0	0	+	+	+	+	0	0	+	0	0
NUMBER THEORETIC	++	++	++	–	++	–	0	++	– –	+	+	– –	– –
SYSTOLIC	++	++	++	–	++	–	0	++	– –	+	+	– –	– –
BROADCAST	+	–	0	0	0	0	+	0	0	0	0	++	0
DATA-DRIVEN	0	0	0	0	0	–	–	++	–	0	+	+	–
ASSOCIATIVE	0	0	0	0	0	0	0	–	–	0	+	++	0

LEGEND
++ HIGHLY SUITED TO APPLICATION
+ WELL SUITED
0 AVERAGE
– UNSUITED
– – HIGHLY UNSUITED

Table II Architecture Capabilities

On the Evaluation of Scene Analysis Algorithms

Kenneth I. Laws, Computer Scientist

Artificial Intelligence Center
SRI International

1. Introduction

This paper describes software evaluation methods developed at SRI International to evaluate contributions to the ARPA/DMA Image Understanding (IU) Testbed. Examples of evaluation results are also presented.

The primary purpose of the IU Testbed is to provide a means for transferring technology from the DARPA-sponsored IU research program to DMA and to other organizations in the defense community. The approach taken to achieve this purpose has two components:

- The establishment of a uniform environment as compatible as practical with the environments of research centers at universities participating in the IU research program. Thus, organizations obtaining copies of the Testbed can receive a continuing flow of new results derived from on-going research.
- The acquisition, integration, testing, and evaluation of selected scene analysis techniques that represent mature examples of generic areas of research activity. These contributions from participants in the IU research program will allow organizations with Testbed copies to begin the immediate exploration of applications of IU technology to problems in automated cartography and other areas of scene analysis.

Evaluation of contributed scene analysis techniques has thus been a major thrust of the Testbed effort. Development of the evaluation methodology has been a related goal. Software evaluation is difficult, and few independent evaluations of IU software have been published. Analysis of an algorithm alone, even if feasible, would neither guarantee correct implementation nor quantify performance on realistic problems. Simple tabulations of pixel classification errors (as in Yasnoff, *et al.* [1]) would not be meaningful for complex scene analysis tasks. Comparative evaluation using several algorithms or software packages on one set of test scenes (as in Ranade and Prewitt [2]) was not practical for testing single algorithms. We have chosen a more subjective approach based on: (1) careful analysis, (2) tests on simple and complex natural scenes, and (3) our own experience in image analysis. This is similar to the method advocated by Nagin, *et al.* [3].

In this paper I describe my experiences with the initial software evaluation efforts on the IU Testbed. I was specifically involved with the evaluation of the GHOUGH object detection system [4] from the University of Rochester, the PHOENIX segmentation system [5] from Carnegie-Mellon University (CMU), and the RELAX relaxation package [6] from the University of Maryland. Many other software packages have been contributed to the Testbed, but have not been as extensively evaluated.

2. Evaluation Purpose

There are many reasons for evaluating software packages. Managers, systems personnel, and users all have different perspectives and different requirements. These imply many different questions that must be answered by a thorough evaluation effort. Some of the major questions are:

- Acquisition — Should the software package be acquired and further evaluated for local applications? What are its capabilities? Can it be extended?
- Implementation — What operating system support is required? How much memory does the package need? How much time does it take to run? Does the implementation correspond to the documented algorithm? Does performance match theoretical predictions? How well is the code structured and commented? Is the documentation adequate?
- Application — Is the package suitable for a particular application? Is the user interface adequate? How does the package perform? Can it be integrated with other packages?

We have attempted to answer these questions in our evaluation reports. The first section of each report introduces the package at a management level, answering questions about the tasks for which the algorithm is suited. Subsequent sections are written for system implementers and for users. The final sections document performance on evaluation tasks and make suggestions for future improvements.

An evaluation effort may have subsidiary effects on the software, the Testbed, and the personnel involved:

- Adaptation — The evaluation effort has spurred several authors to polish or document their software before releasing it to the IU Testbed. Several contributions had to be translated into the C language before submission; the software thus became available on new classes of systems. Such "packaging" can be a significant step in the life of a software system.

- Validation — The processes of translating, installing, and evaluating contributed software have often led to the discovery of programming bugs and occasionally bugs in the algorithms. Where bugs are not found, there is greater assurance that such bugs do not exist.

- Training — We, the evaluating personnel, had to learn to use the software and to understand the theory behind it, thus extending the knowledgeable user community. We have documented this understanding and are otherwise communicating it to others.

- Documentation — Submission of software for evaluation has often spurred initial documentation of the package. Any weakness in this documentation were brought to light as we learned to use the package. We have then filled in the gaps and have added any necessary overview, literature survey, operating instructions, performance examples, and suggestions for improvement.

  We have also placed notes to future implementers and users in the source code and in the on-line **man** page documenting the Testbed version of the contribution. (These **man** pages are included in the Testbed programmer's manual [7].) We believe that such channels of communication between users scattered in space and time are essential for the continued growth of the software.

- Augmentation — We generally had to modify the submitted code to use local graphics and user interface routines, to instrument the code with additional displays or printouts of internal variables, and to rewrite portions of the code to eliminate trivial restrictions or to make the package more efficient for particular tasks. The dividing line between evaluation and new development is not clear, but it is clear that the evaluation effort often leads to improvements in the software. The Testbed environment also had to grow to support the contributions, and many ideas from the contributions have been adapted for use in other software.

3. Evaluation Structure

The tasks involved in evaluating a contribution are reflected in the structure of the evaluation report. We have developed this structure for recording and communicating the results of our investigations.

The introduction to a report summarizes the nature of the reviewed software package, the computer languages or system facilities needed to support it, and the contributions of various people in designing, creating, and maintaining it.

The succeeding background section describes the package from a management viewpoint. Generally this is one of the last sections written because it requires knowledge gained from the entire evaluation effort. First there is a general description of the package, including its purpose, inputs, processing steps, and outputs. Then typical applications and usage scenarios are described, including preconditions and the domain of applicability, relation to preprocessor and postprocessor programs, applications that have been documented in the literature, and potential applications that we or other researchers have suggested.

The background section also describes potential extensions and related applications. Potential extensions are applications that might be feasible if the package were modified or extended, used in a nonstandard fashion, or incorporated as an element of a larger system. Related applications are generalizations or variants of the standard applications for which other techniques seem to be more appropriate.

A descriptive section then documents the algorithm in detail. We begin with its historical development to introduce vocabulary and to put the major technical issues into perspective. Literature references are cited to give credit where credit is due, to aid researchers in finding the full range of concepts that have been explored, and to provide managers and implementors with contacts for further inquiries. The section closes with a detailed statement of the algorithm, including further discussion of design options and references to the literature as appropriate.

The next section is a brief implementer's guide describing the structure of the contributed software and the Testbed locations of its source files, executable files, on-line documentation, and demonstration files. This is information needed to install and run the package or to modify and maintain it. We have included here a description of the SRI modifications to the contribution.

A program documentation section then serves as a users' guide to running the package and invoking all of the algorithm features. We have given instructions for both interactive and batch (or background) execution, including documentation of all command-line invocation options, interactive commands, controlling variables and flags, and status variables. Sometimes we have also found it necessary to give a detailed description of the program's execution phases, complementing the theoretical description of the algorithm in previous sections. This section of the evaluation report could be omitted in cases where existing documents provide adequate and unified documentation of the program.

Our report can now document the evaluation proper. We have divided the evaluation section into two parts: effects of parameter settings and performance statistics for representative tasks. A subjective summary may also be included.

The purpose, intended effect, and legal values for each parameter and control variable were specified in the last section. In this section we probe more deeply, determining the true effect of each parameter on system performance and documenting interactions (either constraints or synergistic effects) with other parameters. The end result is a set of rules for setting the parameters in various processing situations. We also comment on the usefulness of the control features and give suggestions for improving them.

Next we document the performance of the system on selected scene analysis tasks. (Selection of the tasks is discussed later in this paper.) We describe the test protocols, including the input images and the parameter settings that we found optimal for the tasks. We present subjective and objective performance measures and summarize the apparent strengths and weaknesses of the algorithm and the software implementation.

Subjective trials are difficult to document. We ran hundreds of trials on dozens of images. Often a trial designed to investigate one effect would turn up something else as well. It is impractical to illustrate each of these findings in the final report. (Many are of the form "Note how the edge detector found this weak edge but missed that much stronger one.") We have therefore attempted to summarize our findings and present only the relevant information.

In the next report section, we suggest substantial modifications to the algorithm or the implementation. Some of these are of potential, but uncertain, immediate benefit and some are extensions into task areas far beyond those considered by the original author. We also mention known improvements to the contributing institution's continuing software development that have not been incorporated into the more stable Testbed version. Many suggestions are derived from the work of other researchers, in which case we supply the appropriate references. Other suggestions arise from our own evaluation effort.

Our evaluation report concludes with a summary of the major technical concepts and of the strengths and weaknesses of the contributed algorithm and software. Appendices may give further information about the task domain, the algorithm, or the software package that is too detailed for the main body of the report but is not readily available elsewhere.

4. Evaluation Methodology

We have focused our evaluation efforts on the topics of greatest utility. Issues of applicability and of parameter effects and interactions have been given highest priority; issues of resource utilization have been given lower priority, because they are dependent on the algorithm implementation and supporting hardware.

Some of the most difficult evaluation issues have to do with the theory behind the algorithm. We have attempted to summarize the theoretical basis of each contribution, but evaluation of the theory is generally impractical. The best we could do is to document other approaches to similar tasks and to note strengths and weaknesses of the algorithm as reported in the literature or found in our own work. For this reason we have included an extensive literature survey in each of our evaluation efforts.

Fair evaluation of a contribution required that we choose particular tasks for it to perform. Some delicacy was needed in making these choices. It would be unfair to evaluate the software on tasks for which the author considered it unsuited. It would also be pointless to use only tasks that had been well documented in the literature; the essence of evaluation is the learning of something new. We have tried to choose tasks that are well within the contribution's domain and yet of fundamental interest to automated cartography and scene analysis.

The GHOUGH object detection system came with instructions for finding a distinctive lake in an aerial scene. We chose the finding of circular objects in aerial scenes and right-angled corners in oblique scenes as additional tasks. The PHOENIX segmentation system came with a test case of segmenting an orange chair from a white background. We chose skyline analysis as a realistic task. The RELAX probabilistic relaxation system was set up for noise cleaning of an infrared image of a tank. We chose gradient edge detection and segmentation of vehicles from roads as additional tasks. In each case, the imagery was rich enough that performance on auxiliary problems (*e.g.*, nonpurposive segmentation) could be subjectively evaluated.

For each task, we selected suitable imagery, ran dozens of trials to establish optimal parameter settings, and documented the results. If little documentation and operating information came with the package, we spent much of our time learning and recording this information. If documentation was adequate and few parameters had to be explored, we were able to spend more time recording operating characteristics and performance statistics. Economic constraints limited the depth to which any task could be evaluated, but we were able to provide an adequate foundation for future researchers with specific problems.

The first step in evaluating any package was to get it working on a simple test image — usually an image provided by the author. This process occasionally took considerable effort; we chose to rewrite the entire I/O structure and user interface for one program, for instance. This integration effort was essential to the development of the Testbed, but did raise a thorny issue: to what extent should we fix perceived deficiencies and to what extent should we simply document them? One rule of thumb was that we would fix or extend the code in any manner required to carry out an evaluation on realistic tasks.

The next step was to test the software rigorously on one or more simple images. The idea was to become familiar with the workings of the program and with the options available to the user. This step also helped identify software bugs or misunderstandings about the intended functions of the program. We strongly recommend the use of generated or well understood problems as one phase of the evaluation effort.

Investigation of parameter interactions was one of the most difficult evaluation tasks. Analysis of simple imagery permitted us to concentrate on internal variables instead of interactions with a complex environment. Even so, the "search space" of possible interactions was immense. The PHOENIX program, for example, has 14 major threshold values to control image segmentation, in addition to various control strategies and interaction options.

One could navigate this complexity by using an intelligent driver system to monitor thousands of runs, modifying parameter settings each time to optimize some performance criterion. While such an approach is feasible [8], it would have provided only a superficial understanding of *why* the identified parameter sets were optimal combinations. We chose instead to analyze the program structure, experiment with carefully chosen parameter values, and study the execution (as opposed to a single result) of each computer run. Often we had to disable features of the program in order to study one feature in isolation.

The final experimental step was to evaluate the software for realistic tasks on "natural" imagery. This proved to be exceedingly difficult because the space of input imagery was impossibly large. If a program could detect circular tanks in one image, for instance, would it be able to detect them at different image resolutions? With different levels of contrast and blur? With strong shadows and highlighting present? With occlusions, unusual edge alignments, or texture effects present? Would it be able to distinguish real targets (possibly camouflaged) from decoys and destroyed targets?

Such questions go beyond the scope of this initial evaluation effort. We tried our best, however, to get a "feel" for each program's capabilities. We varied pertinent imagery variables and carefully noted the effects. Anomalies were checked out by instrumenting the code or by experimentation on simple images. We believe that this intelligent experimentation is at least as useful as extensive statistical validation would be.

Unfortunately we were not able to devise rigorous performance metrics for tasks such as "target detection." We carefully tuned the analysis system for each problem and reported the best performance that could be obtained. (We tried to avoid tuning the system for each image, however. A single parameter set or operating procedure was developed for each task.) The results are necessarily subjective and would vary slightly for other tasks, other imagery, or other experimenters.

5. Evaluation Examples

It is difficult to convey the scope and variety of our evaluation results in a short paper. The PHOENIX report alone is more than 80 pages long, with 25 pages devoted to performance evaluation and suggestions for future development. I will illustrate the evaluation results by presenting short excerpts from the reports. Different reports will be used to illustrate different points about the nature of the evaluation effort.

5.1. Theory

We have documented the theoretical basis and the implementation of each contribution from several perspectives. We have provided theoretical justifications and mathematical notations where appropriate, and have then related this information to the parameters and commands of the software packages. Sometimes it was quite difficult to extract this information from the technical literature.

The RELAX system, for instance, could be regarded as a general method for local modification of constraint and compatibility information stored in the nodes of a rectilinear graph. The initial label probabilities at the nodes may be derived from image pixel intensities and the final label assignments may be mapped back to pixel intensities, but the iterative relaxation technique is independent of image-domain considerations. Much of the theoretical work on relaxation has abandoned the rectilinear image plane and has dealt with constraint relations on arbitrary graphs with varying numbers of neighbors for each node.

For the evaluation, we extracted the updating equations actually used in the software package and expressed them in terms common in the theoretical literature. The RELAX package includes both the Hummel-Zucker-Rosenfeld additive updating scheme and the Peleg multiplicative updating scheme. Here is part of our description of the former.

> The goal of the relaxation algorithm is to update the values of the probabilities associated with a node to reflect the compatibility of neighboring labels. The $(t+1)$ update of the kth label value is calculated from the previous time (t) update by
>
> $$q_i^{(t)}(\lambda_k) = \frac{1}{m} \sum_j \left\{ \sum_{\lambda'=\lambda_1}^{\lambda_n} r_{ij}(\lambda_k, \lambda') p_j^{(t)}(\lambda') \right\}$$
>
> $$p_i^{(t+1)}(\lambda_k) = \frac{p_i^{(t)}(\lambda_k)\left(1 + q_i^{(t)}(\lambda_k)\right)}{\sum_{\lambda=\lambda_1}^{\lambda_n} p_i^{(t)}(\lambda)\left(1 + q_i^{(t)}(\lambda)\right)}$$
>
> where j indexes the m neighbors of node i. $r_{ij}(\lambda_k, \lambda')$ is the compatibility coefficient for node i with label λ_k and neighboring node j having label λ'. $q_i(\lambda_k)$ can be though of as the assessment by the neighboring nodes that node i should be labeled λ_k, while $p_i(\lambda_k)$ is the assessment by node i as to its own label. These two assessments are combined to produce an updated probability, $p_i(\lambda_k)$.
>
> The compatibility coefficients may be negative if the labels are incompatible, positive if the labels are compatible, and zero if they are independent. While it is possible to define the compatibility coefficients in terms of conditional probabilities, it is overly restrictive to do so. The compatibility coefficients for the Hummel-Zucker-Rosenfeld rule are based on information theory; mutual information defines the compatibility coefficients and provides a mechanism for calculating them:

$$r_j(\lambda_k, \lambda_l) = \frac{1}{5} \ln \frac{w \sum_i p_i(\lambda_k) p_{i_j}(\lambda_l)}{\sum_i p_i(\lambda_k) \cdot \sum_i p_i(\lambda_l)}$$

where k and l range over the n labels, i ranges over all w nodes of the graph, and j specifies the particular neighbor (*e.g.*, upper-left) of the i-th node. For each node i, $r_{ij}(\lambda_k, \lambda_l)$ is equal to $r_j(\lambda_k, \lambda_l)$ clipped to be in the closed interval [-1,1].

In the report we have discussed the meaning of these terms and methods of estimating or setting the numeric values, as well as the effects of relaxation in various image analysis tasks.

5.2. Analyses

The following are a few results from the performance analyses on the PHOENIX and GHOUGH systems. Space limitations prevent a full description of all of the terms used, but the examples should give some feeling for the level of understanding that a thorough evaluation may require

The PHOENIX segmenter is a moderately complex system with 14 user-settable variables that control the segmentation process itself. The original contribution came with very little guidance about setting these parameters to achieve reasonable segmentations. One of our evaluation tasks was the creation of such information.

We began by finding a set of parameter values that would segment simple scenes of large objects down to the level of major subregions. We called this a "moderate" segmentation. Then we developed a set of parameter values for very coarse segmentation using "strict" heuristics to disallow most potential region splits. Finally we developed a set of values for complete or overly permissive segmentation using "mild" threshold screening heuristics.

Each column in Table 1 lists one of these parameter sets. The user need only select the extent of segmentation desired and load the corresponding parameters. We thus reduced the 14 parameters to a manageable single decision that is relatively independent of the image content. Additional flexibility is possible by switching parameter sets during a segmentation run to control the fineness of segmentation within particular image regions.

It will occasionally be necessary for the user to deviate from the recommended parameter settings. To make this possible, we have evaluated each parameter individually. Here is part of the **maxmin** parameter description:

> **Maxmin** is the minimum acceptable ratio of apex height to higher shoulder for an interval in the histogram. Any interval failing this test is merged with the neighbor on the side of the higher shoulder. The test is then repeated on the combined interval. The overall effect on a set of cutpoints is to eliminate those that are on the sides or tops of major peaks.
>
> **Maxmin** is a powerful heuristic. With strict smoothing and all other heuristics disabled, **maxmin** alone is able to produce reasonable segmentations. It is even more powerful when combined with the area heuristics. With mild or moderate smoothing, **maxmin** passes clusters of cutpoints in the noise regions between major peaks. This is fine if the clusters can be thinned by the **absarea** and **relarea** heuristics, but a poor selection may be made if they are left for the **intsmax** heuristic.

Parameter	Strict	Mod.	Mild
depth	4	10	20
splitmin	200	40	20
hsmooth	25	9	5
maxmin	300	160	130
absarea	100	30	5
relarea	10	2	1
height	50	20	10
absmin	2	10	30
intsmax	2	3	6
isetsmax	2	3	5
absscore	925	700	600
relseore	95	80	65
noise	50	10	5
retain	4	20	40

Table 1: PHOENIX Segmentation Parameter Sets

The problem here is that PHOENIX has no "quality" score for histogram valleys. It assumes that cutpoint bin height is an adequate measure, whereas width and depth relative to the neighboring peaks are also important. PHOENIX can only incorporate such knowledge by smoothing the histogram, and the amount of smoothing required depends on how separated the peaks are.

The next step in the PHOENIX evaluation was investigation of a skyline delineation task. One of the test images, *portland*, shows a city skyline against a cloudy sky. After describing segmentation performance on reduced versions of this and other images, we reported the following:

> A test sequence was run on the full-resolution (500x500) *portland* image. Strict and even moderate heuristics were unable to segment the image when only the red, green, and blue feature planes were used; it was necessary to use the mild heuristics. The best approach would be to start the segmentation with mild thresholds and then return to strict or moderate ones for segmenting the subregions. Instead, we avoided such special interference and ran the segmentation to completion using mild heuristics. The full run (which, with the 'v' flag set, generated 19,000 lines of printout) required 33 minutes of CPU time:

PHASE	REAL	CPU
Histogram	0:04:13	0:02:32
Interval	0:18:12	0:07:27
Threshold	0:10:00	0:03:47
Patch	0:03:51	0:03:30
Collect	0:38:12	0:14:04
Segmentation	1:18:15	0:32:34

The final segmentation into 1182 regions (including

> nearly every window of every building) was much better than the original attempt, but still had difficulties distinguishing a glass-surfaced building from the sky that it reflected.

Later test runs showed even better performance when color transforms were used in addition to the three original color planes. Based on our experience with these tests, we were able to suggest operating procedures for the use of PHOENIX in similar tasks.

The evaluation of the GHOUGH object detection system was similar. Because GHOUGH had fewer parameters, we were able to spend more time analyzing system performance on realistic tasks. One thrust of this effort was to develop an understanding of specific operational characteristics, as in the paragraphs below.

> The requirement of sharp edges does not imply that smooth, continuous object boundaries are required. The program is quite tolerant of noise in the outline and is able to find irregular, incomplete, or discontinuous shapes. The circle template, for instance, often responds to forest clearings, tree tops, road intersections, and curved embankments, as well as to square buildings and to image "hot spots." The irregularities in these image structures spread the vote cluster in the accumulator, but the local maximum may still be above the general noise level.
>
> Shadow edges usually fit the requirement for strong, sharp edges. It is often easier to find a shadow than to find the object that cast it. This may be a useful cueing technique, but must be used carefully to avoid reporting objects at incorrect locations. A similar problem exists with high-resolution imagery: the position reported for a part of an object (*e.g.*, the circular top of a storage tank) may not correspond to the position of the whole object.
>
> These characteristics mean that the program is best suited for three tasks: locating industrial parts in high-contrast imagery; counting numerous, obvious, similar objects such as storage tanks, barracks, or microscopic particles; and precisely positioning a template when an approximate location is cued by the user or by another system. Even for these applications, the program must be supervised and its output edited. Other applications will require further development of the technique.

Sometimes our results were quite unexpected, as when we found that increasing the number of points in the search template definition had no effect on execution time and could actually decrease target location accuracy. Execution time was unaffected because each edge in the image votes for only the best matching template edge (or set of edges), regardless of the number of similar or nearby edges in the template. Performance could be degraded because the template points were entered at discrete points on a Cartesian grid, and close spacing of the points caused severe quantization of the relationships between them.

We were able to quantify system performance on representative tasks. Some of the domain-independent equations are given below. (The terms are fully explained in the evaluation report.) We have attempted to base the formulas on important characteristics of the GHOUGH algorithm, although the coefficients had to be estimated empirically.

$$\begin{aligned} \textit{Edge time} &= .00036(\textit{window points}) + .0053(\textit{edges found}) \\ &\quad + .00019(\textit{accumulator entries}) + (\textit{additional paging time}) \end{aligned}$$

$$\begin{aligned} \textit{Analysis time} &= 10^{-4}(\textit{search volume}) \\ &\quad \times (.08 + 2.6\log(1 + \textit{accumulator density})) \\ &\quad + (\textit{additional paging time}) + .0025(\textit{maxima found}) \end{aligned}$$

$$\textit{Maxima} = .023(\textit{search volume})^{0.8}(1 + \textit{accumulator density})^{2} - 1$$

$$\textit{Noise} = 2.04(\textit{search volume})^{0.09}(1 + \textit{accumulator density})^{0.8} - 1$$

Such formulas would be very helpful in designing an improved version of GHOUGH. Even more exciting is the possibility of building an expert image analysis system that would include GHOUGH as a component. The knowledge base of such a system would record predictive formulas and other operating characteristics in a form that could be used by both humans and machines.

Some of the GHOUGH parameters are dependent on image content. These were very difficult to quantify, but we attempted to document the dependencies well enough that users could adapt our findings to their own imagery. The following is our discussion of GHOUGH performance as a function of the edge-detection threshold.

> The number and density of edges detected in an image are sigmoid (s-shaped) functions of edge threshold similar to cumulative frequency histograms. GHOUGH operates best when 10% to 20% of the pixels are classified as edge points, although it will usually work well at any edge density above 6%. Some typical threshold values to achieve specified edge densities are:
>
> | Scene Type | 6% | 12% | 25% | 50% |
> |---|---|---|---|---|
> | Cloudy sky | 42 | 35 | 28 | 20 |
> | Aerial terrain | 160 | 120 | 80 | 40 |
> | Aerial target area | 200 | 180 | 120 | 60 |
> | Low-angle urban | 260 | 200 | 140 | 90 |
> | Forest cover | 230 | 220 | 160 | 100 |
> | Aerial urban | 720 | 600 | 480 | 340 |
>
> In general it is better to use too low a threshold: this will increase chances of finding target edges while only slightly increasing noise level, and the edges found are likely to be the most reliable ones. The main drawback is that low thresholds increase the time required to fill the accumulator with votes. A reasonable starting guess is a threshold of 120.

As we experimented with the software packages, we noted a great many characteristics that could be improved. The preceding GHOUGH edge-threshold sensitivity, for instance, led us to suggest that an adaptive edge detector be used. Our suggestions have covered everything from the algorithm to the characteristics of larger systems that might incorporate these routines.

5.3. Summaries

We have also tried to summarize our findings, drawing on our experience with other image analysis systems. Each of the reports ends with such a summary. For the PHOENIX system, our observations included the following:

> The PHOENIX segmentation system is one of several existing systems for recursively segmenting digital images. Its major contributions are the optional use of multiple thresholds, spatial analysis for choosing between good features, and a sophisticated control interface. Some of the strengths and weaknesses of the PHOENIX algorithm are listed below.
>
> PHOENIX, like other region-based methods, always yields closed region boundaries. This is not true of edge-based feature extraction methods, with the possible exception of boundary following and zero-crossing detection. Closed boundaries are the essence of segmentation and greatly simplify certain classification and mensuration tasks.
>
> PHOENIX is a hierarchical or recursive segmenter, which means that even a partial segmentation may be useful. This can save a great deal of computation if efforts are concentrated on those regions where further segmentation is critical. If PHOENIX is to be driven to its limits, other methods of segmenting to small, homogeneous regions may be more economical.
>
> PHOENIX is relatively insensitive to noise. Thresholds are determined by the feature histograms, where noise tends to average out. This contrasts with edge-based methods, where the local image characteristics can be highly perturbed by noise.
>
> PHOENIX has no notion of boundary straightness or smoothness. This may be good or bad depending on the scene characteristics and the analysis task. It easily extracts large homogeneous regions that may be adjacent to detailed, irregular regions (*e.g.*, lakes adjacent to dock areas or sky above a city); such tasks can be difficult for edge-based segmenters.
>
> PHOENIX tends to miss small regions within large ones because they contribute so little to the composite histogram. It is thus poorly suited for detecting vehicles and small buildings in aerial scenes, although there may be ways to adapt it to this use. It also tends to misplace the boundary between a large region and a small one, thus obscuring roads, rivers, and other thin regions. Boundaries found by edge-based methods are less affected by distant scene properties.
>
> PHOENIX may also fail to detect even long and highly-visible boundaries between two similar regions if the region textures cause their histograms to overlap. Edge-based methods are better able to detect local variations at the boundary.
>
> Since perfect segmentation is undefined, PHOENIX must oversegment an image in order to find all region boundaries that may be of use to any higher-level process. It is left for a segmentation editing step to merge segments that have no usefulness for some particular purpose. Without having such a step, or indeed even a purpose, it is very difficult to evaluate the segmenter output.

6. Conclusions

Our evaluation efforts have documented a great many suggestions for improving the evaluated software. We have tried to be as quantitative and rigorous as possible, but the results are necessarily subjective. Often we have functioned as restaurant or theater critics do, reporting our impressions of the contributions. These informed opinions, combined with our more rigorous documentation, should provide a good basis for more specific evaluation efforts directed at particular task scenarios and production environments. Our evaluation reports and the SRI Testbed environment make the contributed programs available as benchmark systems and as research tools.

REFERENCES

1. W.A. Yasnoff, J.K. Mui, and J.W. Bacus, "Error Measures for Scene Segmentations," *Pattern Recognition*, Vol. 9, pp. 217-231, 1977.
2. S. Ranade and J.M.S. Prewitt, "A Comparison of Some Segmentation Algorithms for Cytology," *Proc. 5th Int. Jnt. Conf. on Pattern Recognition*, Miami Beach, FL, pp. 561-564, Dec. 1980.
3. P. Nagin, R. Kohler, A. Hanson, and E. Riseman, "Segmentation, Evaluation, and Natural Scenes," *Proc. IEEE Conf. on Pattern Recognition and Image Processing*, Chicago, pp. 515-522, Aug. 1979.
4. K.I. Laws, *The GHOUGH Generalized Hough Transform Package: Description and Evaluation*, Technical Note, Artificial Intelligence Center, SRI International, in press.
5. K.I. Laws, *The PHOENIX Image Segmentation System: Description and Evaluation*, Technical Note, Artificial Intelligence Center, SRI International, in press.
6. K.I. Laws and G.B. Smith, *The RELAX Image Relaxation System: Description and Evaluation*, Technical Note, Artificial Intelligence Center, SRI International, in press.
7. K.I. Laws, *The DARPA/DMA Image Understanding Testbed Programmer's Manual*, Technical Note, Artificial Intelligence Center, SRI International, in press.

8. D.B. Lenat, W.R. Sutherland, and J. Gibbons, "Heuristic Search for New Microcircuit Structures: An Application of Artificial Intelligence," *The AI Magazine*, Vol. 3, No. 3, pp. 17-33, Summer 1982.

IMAGE UNDERSTANDING APPLICATION PROJECT: IMPLEMENTATION PROGRESS REPORT

G.R. Edwards, F.M. Vilnrotter,
D.M. Keirsey, B.L. Bullock, D.Y. Tseng
HUGHES RESEARCH LABORATORIES
Malibu, California 90265
and
D.H. Close, J.F. Bogdanowicz, E.P. Preyss,
H.A. Parks, D.R. Partridge
SPACE AND STRATEGIC SYSTEMS GROUP
HUGHES AIRCRAFT COMPANY
El Segundo, California

ABSTRACT

This paper reviews the current status of an ongoing program to demonstrate the application of DARPA Image Understanding research to a photo interpretation system using real imagery. The vision system developed thus far is based on the Acronym vision system, developed by Rod Brooks and Tom Binford at Stanford University on the DARPA Image Understanding Project. This system has provided the basis for a sophisticated vision capability, implemented on a general purpose computer. In particular, the system includes symbolic prediction of shapes from both oblique and vertical camera views, prediction of shadow shapes, a model-driven shape extraction system, area of interest operator, and a situation assessment module to perform temporal analysis. Having recently completed and exercised the first system implementation, this discussion will include both details of system design and issues related to performance and future extension.

1 - OVERVIEW

Hughes, with support from DARPA and ONR, is conducting a program to apply research results from the DARPA Image Understanding Project [7]. This program has combined several existing image understanding components, along with new approaches, to address problems associated with automated photo interpretation.

The system which has been developed attempts to identify interesting objects by matching shapes extracted from digitized images to shapes generated by geometric analysis of three-dimensional object models and information describing the camera and illumination conditions. Low-level processing provides identification of likely areas of interest in each scene, and edge-based lines extracted from these regions are provided as input for shape extraction. Descriptions of predicted shapes, representing both visible portions of object sub-parts and shadows cast from these sub-parts, are also provided to direct the shape extraction process. Extracted and predicted shapes are finally compared, and matching shape dimensions and spatial relationships are used to determine instances of both generic objects and specific modeled sub-classes. In addition, object identifications through a sequence of images may be interpreted by a script-based situation assessment module which is capable of improving object identifications and making inferences about the actions of objects.

This paper will describe the current vision system and situation assessment implementations and summarize areas where additional extension is necessary. In Section 2, specific system modules will be described, both in terms of approach and ultimate performance when applied to typical application imagery. Plans for extension of this system in an anticipated second contract phase will be discussed in Section 3.

2 - SYSTEM COMPONENTS

The vision system structure for the image understanding system has been provided by the Acronym system [3-6], developed by Rod Brooks and Tom Binford at Stanford University on the DARPA Image Understanding Project. This system provided a powerful set of building blocks, including a rule system, slot / filler style record package, and a constraint manipulation system supported by an algebraic simplifier. Higher level Acronym modules provided vision system components which, although requiring substantial extension by both Hughes staff members and by Dr. Brooks to support a more general image understanding capability, formed a powerful working vision system structure.

In addition to the capabilities provided by the Acronym system, additional components were added to provide contrast enhancement, area of interest identification, image scaling, line extraction, and temporal situation assessment.

The remainder of this section will describe each portion of the image understanding vision system in more detail, including comments about the actual performance of each module when applied to typical application imagery. The entire system has been implemented on a VAX 11/780 general purpose computer, utilizing the VMS operating system, Eunice (a Unix emulation package), and the Franzlisp lisp implementation. Selected low level processing steps were implemented in the Fortran or C programming languages. Mean filter operations were performed utilizing a DeAnza IP 8500 Video Display Processing unit.

2.1 - PRE-PROCESSING

Image digitization techniques may result in distorted object dimensions caused by oversampling or undersampling of the original photograph during the digitization process. Since dimensional information is of key importance in the operation of the vision system, utilities have been provided to rescale the digital data using bi-linear interpolation techniques. Interpolation may result in significant edge degradation, manifested as broken representations of otherwise dominant and continuous edges. This effect is minimized by the application of a mean filter, typically using three by three or five by five masks, depending on the range of interpolation.

2.2 - AREA OF INTEREST OPERATOR

Determination of interest areas in a scene is approached by determining whether local sixteen by sixteen pixel regions in an image represent a portion of an object or background. Masks, sized appropriately for the objects of interest in a given application and for a limited range of image resolution, are applied across the image at fifteen degree incremental rotations. Each area thus measured is "scored" according to the number of object points contained within the area. Areas with a sufficiently high score are then deemed "interesting" and later serve to restrict the search area for the remainder of the vision system.

In order to identify object points, the image data is first contrast enhanced using a three part piecewise linear lookup table, where the points of discontinuity are determined manually by heuristics driven by the image histogram. Successive median filters and an optional Nagao smoothing operator are applied to significantly smooth intensity edges. This enhanced image is then processed to determine local mean and standard deviation values within sixteen by sixteen non-overlapping blocks. The determination of object vs. background for each block is based an the derivative of these measures, using a constant decision surface determined by training over a range of test imagery related to the application.

Several comments can be made regarding the performance of this area of interest operator. First, the technique seems well suited to the current application, where background intensities tend to be fairly homogeneous and uncluttered. The algorithm was able to consistently identify object areas in imagery and reject clutter. However, it is believed that applications where the background is likely to be highly cluttered may require a more complex set of discriminant values and a more carefully trained decision surface.

2.3 - LINE EXTRACTION

The extraction of lines from imagery is of key importance, since these lines provide the sole information from which observed shape descriptions are determined in this system. In the current implementation, the Nevatia / Babu linefinding algorithm has been implemented in the C language, with some modifications to the bridging and linking algorithms. This linefinding system has proven a valuable means of extracting most necessary edges separating object subparts from background. However, two specific problem areas have been identified which will ultimately require a more specialized approach to edge identification.

The first range of segmentation problems involve edges which are easily discerned by human observers, but which fail to be extracted by the Nevatia / Babu algorithm. Typically, these edges are not characterized by a significant intensity gradient, but rather the border of a significant change in texture regions. A second set of problems (often overlapping with the texture border problem) are related to the extraction of very soft shadow boundaries. In these cases, the intensity gradient across the boundary, although consistent across the entire shadow edge, is of such a small magnitude as to be inseparable from background clutter. Reduction of edge thresholds to identify these boundaries would result in an unmanageable number of insignificant edges being identified.

2.4 - OBJECT MODELING

Interesting objects are modeled by the Acronym modeling system by three dimensional models built from generalized cylinder volumetric elements [1]. These volume primitives are represented in general by a cross sectional face which describes the volume as it is swept along an axial spine, with the dimensions of the face being varied along the axial sweep. In practice, limits imposed by the complexity of performing geometric reasoning later in the vision system limits the modeling package to the use of circular and rectangular faces swept along straight spines.

Dimensions of objects and affixments between subparts may be described as constrained ranges of values. Further, loosely constrained descriptions may be used to specify generic object classes, while more tightly constrained descriptions may be specified in parallel to describe subclass specializations.

This modeling capability, along with an interactive graphics interface, was provided intact within the Acronym system. Issues of performance, in terms of the ability to represent objects with sufficient accuracy, will be discussed below.

2.5 - SHAPE PREDICTION

In the prediction module, each three-dimensional volume element in the object model is used to generate a set of two-dimensional shapes which represent the visible end faces and swept surfaces of the volume as seen from the modeled camera position. The predicted 2-D shapes are represented by ribbons (rectangular shapes).

Spatial relationships are described by arcs linking these shapes in specific manners, for example angle arcs and distance arcs.

The original Acronym system provided the necessary geometric reasoning knowledge to provide two dimensional descriptions of shapes which represent the three dimensional object subpart mudels as seen from a modeled vertical camera. However, two major extensions were required to provide the necessary shape prediction capabilities for a more general vision system capability.

To begin, it was necessary to enhance support for the prediction of shapes from an oblique camera view. The prediction process functions by building symbolic expressions describing the transformations between object, camera, and world coordinate systems for the model faces determined to be visible from the camera view. These expressions grow considerably more complex in the case of an oblique camera view, and support for the simplification of these expressions required major extensions to the geometric simplification module originally contained within the Acronym system.

As now implemented, the prediction module is capable of predicting both object subpart shapes and their relative spatial relationships for both vertical and oblique camera views. This effort was supported greatly both by consultation with Dr. Brooks, and by the example of the originally coded vertical camera case. The resulting system is limited only in that modeled object positions and orientations relative to the camera line of sight must be specified exactly for the oblique camera case. This limitation is overcome in operation by operating the vision system in two passes, the first of which applies an overhead camera approximation to loosely constrained models. This results in the identification of candidate matches, which provide specific positions and orientations for performing more detailed predictions using detailed models and a fixed, oblique camera model.

The second major extension was the addition of a rudimentary capability to predict shadows cast by object subparts from a known illumination direction. Most research in this area has been directed toward the bottom-up interpretation of shadows to gain insight about the object casting the shadow. Within the scope of the image understanding vision system, an assumption has been made that we have accurate representations of interesting objects. Also, there is no a priori knowledge of what observed image shapes represent cast shadows. In keeping with these considerations, the system design was extended to include the prediction of shadow shapes from modeled object subparts in a manner analogous to the prediction of object shapes.

The Acronym prediction system was extended to include shadow prediction for a limited set of objects. Shadow prediction was designed and implemented for two cases, a right circular cylinder with its spine parallel to the shadow plane and a rectangular parallelepiped with one of its axes perpendicular to the shadow plane.

Given a world coordinate system, in which a camera, an illumination source, a shadow plane, and a solid object are defined, Acronym attempts to predict the dimensions of the shadow cast on the shadow plane as they will appear in the image. This shadow plane is the xy plane in the world coordinate system. The camera and the object, as well as each of its subparts (cones), have their own coordinate systems.

The first part of the shadow algorithm deals with determining the rotation expression necessary to perform transformations between the cone and world coordinate systems and the cone and camera coordinate systems. Most of the computations occurring in the shadow module are performed in the object (cone) coordinate system. The shadow plane equation, the illumination direction vector, and the camera line of sight vector are all calculated with respect to the object cone. Then some tests are performed to insure that neither the camera line of sight nor the illumination vector is parallel to the spine of the object cone (if the object cone is a cylinder), and that the illumination direction is not parallel to the x, y, or z axis (if the object is a rectangular parallelepiped). This is the point where the algorithm becomes specialized as to particular object type.

There are two cases presently handled by Acronym. They are a right circular cylinder with its spine parallel to the shadow plane, and a rectangular parallelepiped with one of its axes perpendicular to the shadow plane.

Using the illumination direction defined with respect to the coordinate system of a solid object, it can be determined whether or not a particular planar surface or face on that object is illuminated. This is easily done by examining the dot product of the surface normal and the illumination direction vector. A negative result implies that the planar surface is illuminated, while a positive result implies that the planar surface is not illuminated. In the case of a curved surface the location of the illumination boundary can be calculated. That is, the points on the surface where the illumination vector is perpendicular to the surface normal can be determined. This set of points forms the dividing line between the illuminated and shadowed subcontours of the surface. In the case of the rectangular parallelepiped the shadowed sides are found, while the cylinder case requires that the illumination boundary be calculated. These shadowed faces and subcontours need to be determined in order to properly predict the shadow cast by the object.

Initially it was thought that undistorted shadow contours would be able to be predicted along with their spatial relations and that the appropriate shape distortions caused by the camera angle could be anticipated. This approach would have paralleled the Acronym algorithm quite closely. However, it was found that in order that the subcontours of the cylinder be correctly predicted, a new algorithm needed to be adopted which would include projecting ribbon vertices onto the image plane and finding the best rectangular fit approximating the actual ribbon dimensions. Therefore, instead of defining a contour corresponding to a surface and predicting how its dimensions will change given a set of camera coordinates, the contour nodes and dimensions in the image plane are directly calculated. This is done in the case of the right circular cylinder; however, the case of the rectangular parallelepiped is handled differently.

The dimensions of the ribbons corresponding to the rectangular solid are predicted by the original acronym method. Its cast shadow, however, is handled by a separated routine. First the apparent length (in camera coordinates) of the shadow cast by a unit vector parallel to the object spine is calculated. Next, bounds for the actual shadow length are calculated in two ways:

- First, by casting this shadow onto the shadow plane, using the perpendicular length from the shadow plane to the top of the rectangular solid and adding this perpendicular length to the length of the shadow which is cast; and

- Second, by casting this shadow onto the plane parallel to the shadow plane through the base of the rectangular solid, using the solid's vertical length.

The width is calculated by using a normalized, rotated, and scaled version of the cast shadow vector.

The right circular cylinder can give rise to a maximum of 6 ribbons:

- S1 - the ribbon representing the entire curved surface of the cylinder. It can contain 2 subribbons, S2 and S3.

- S2 - the illuminated part of the cylinder's surface.

- S3 - the self-shadowed part of the cylinder's surface.

- S4 - the shadow cast onto the shadow plane by the swept contour of the cylinder.

- S5 - the entire shadowed area including S3 and S4.

- S6 - the entire area covered by the cylinder and its cast shadow. This ribbon contains subribbons S1 through S5.

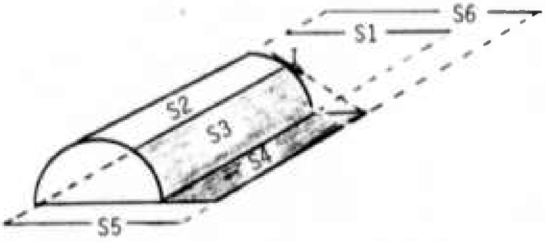


Figure 1. Predicted Right Circular Cylinder Ribbons

The rectangular parallelepiped can give rise to a maximum of 4 ribbons:

- S1 - the ribbon representing the one or two visible swept planar surfaces of the rectangular parallelepiped.

- S2 and S3 - The two visible planar surfaces of the rectangular parallelepiped.

- S4 - The shadow cast onto the shadow plane by the swept contour of the rectangular parallelepiped.

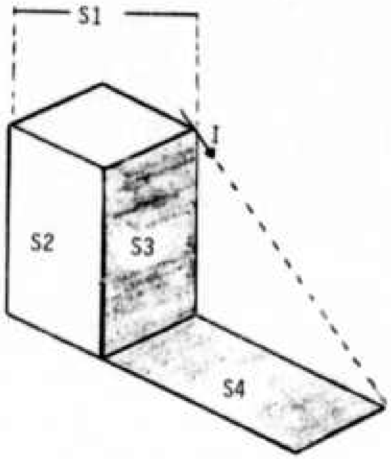


Figure 2. Predicted Rectangular Parallelepiped Ribbons

Only the S4 ribbon, the rectangular solid's cast shadow, is predicted in the shadow prediction module, while all six of the cylinder's ribbons are predicted here. New rules were added to some of the core Acronym rule sets such as the spatial relation and interpretation rule sets to accommodate the addition of this new set of ribbons.

2.6 - SHAPE EXTRACTION

In order to provide a sufficiently robust shape extraction capability, a new module for the identification of rectangular shapes, or ribbons, was implemented to replace the original Acronym ribbon finder [2]. This new design takes a top-down approach to shape extraction, using the predicted shape dimensions to drive the selection and operation of rule-based heuristics which build shape descriptions from extracted line segments.

The new ribbon finding module makes use of the results of the area of interest operator, rejecting line segments which lie outside these interesting regions. The surviving line segments are then used to identify various line-based image features, including parallel lines, colinear lines, and line pairs which form vertices. Since the identification of these features is computationally expensive, a library of these features is built before applying the various ribbon finding heuristics, to avoid recomputing similar features as each heuristic rule set searches for particular shapes. In fact, well over half the CPU time for a typical ribbon finding application is spent identifying these features.

Having created a library of these features, each predicted shape is submitted to a set of rules which in turn applies appropriate heuristic rule sets which attempt to identify instances of that shape from the line segments and line based features available. For example, a long predicted shape would trigger the invocation of rule sets which search for parallel lines or a long segment corresponding to the dominant long sides of the shape. These heuristics make use of the predicted shape dimensions to search, for example, for parallel sides whose separation falls within the constrained range of width for the predicted shape. Similarly, features which indicate the end closings of the shape are tested for satisfaction of the predicted range of length.

Separate heuristic rule sets are invoked by the selection rules depending on the type of shapes predicted. Long shapes are identified, as exemplified, by prominent side features, while more square shapes are identified by either prominent sides or ends, or by corners. Small shapes are more typically described by a sets of segments which completely close the shape, and rules are implemented for this case as well. Finally, shadow shapes are often supported by only line features, and heuristics searching for any one of the four sides are included. As each rule set builds appropriate ribbon shapes, they are appended to a graph structure to provide access by the matching and scene interpretation module.

In practice on application imagery, the success of this model directed shape extraction system is dependent on two related modules. First, the low level processing and line finding must be capable of providing sufficient quality of features to support the heuristic extraction of shapes. Secondly, there is the assumption that the chain of approximations which model object subparts as generalized cylinders, and predicts them as ribbon shapes, adequately represents the actual shapes to be found in the image data. If this is not the case, the model based predicted shapes actually misdirect the ribbon finder. A serious instance of the problem occurs in the prediction and extraction of shadow shapes. To begin, shadows are approximated as rectangular ribbons, when the actual calculated ribbon would be more suitably represented as a parallelogram. Since shadow boundaries are typically weak, and the ribbon finding heuristics thus rely on very few features, it is not uncommon for the resulting extracted ribbon to be misoriented by the angular error resulting from the rectangular approximation. In addition, the predicted width of the rectangular shape is taken along the radius perpendicular to the spine of the ribbon, while the parallelogram shape actually seen in the image data will have end segments which are not perpendicular to the sides or spine, and are longer than the predicted width.

Apart from these problems the ribbon finder has performed quite well on shapes of varying resolution and with limited edge features describing the important shapes. Performance is greatly enhanced, especially in terms of eliminating clutter shapes, if the predicted shapes are tightly constrained, implying tight model constraints and an accurate camera model.

2.7 - INTERPRETATION

The interpretation module performs two broad functions within the vision system. First, it performs a matching function, locating sets of observed shapes which are consistent with a set of predicted shapes, both in terms of shape dimensions and relative spatial relationships. Secondly, these match sets are interpreted by determining what modeled generic class or subclass specializations are satisfied by each match set.

The first function, the creation of match sets, builds sets of nodes which contain the predicted and extracted shapes which match dimensionally. Graphs are built which link these nodes using arcs which describe the spatial relationships between them, driven by the predicted spatial arcs. Further, as these graphs are built, corresponding restriction nodes are constructed which describe the shape and arc matches in terms of the model and camera parameters, forming back constraints which restrict the interpretation of each match. The second interpretation function determines the satisfiability of these back constraints, thereby determining membership of the matched object in specific modeled classes. These interpreted graphs form the final result of the vision system as applied to a single scene.

As in the prediction module, the interpretation system required significant extension, especially to support the matching of shadow shapes. These shadow cases required a significant recoding to support new spatial arcs, as well as to improve the performance of the matching and interpretation tasks.

2.8 - SITUATION ASSESSMENT

In many applications, models of temporal activities of various interesting objects can provide the overall system with capabilities beyond those achievable with a single-frame vision system alone. These applications involve situations where the objects have predictable behavior patterns. More specifically, these applications involve objects whose observed locations over a sequence of images may imply their behavior, and the seouences of observed activities form a predictable history.

Tracking observed objects may serve to confirm or disconfirm the results of the vision system object classification. If the vision system misclassifies an object, then the situation assessment module has the opportunity to detect the error by either noticing the change in an object identification of a previously known and tracked object, or by detecting an object not behaving in a normal manner. These situations may be flagged or corrected, depending on the confidence or ambiguity in the inferred identification.

The method implemented to monitor object behavior is based on the concept of a knowledge structure which describes expected behaviors and lengths of activities called scripts. These scripts are matched against observed object identification sequences to infer activities. Inclusion of knowledge of activity durations allows further inference even when some stage of the script may not have been observed.

As the script activities are tracked, various ambiguities may make it necessary to carry along multiple script interpretations, which are weeded out as more observations are made, until either a consistent interpretation is found or all the interpretations are eliminated, at which time an anomalous situation would be flagged. This script processing system has been developed and exercised using manual observation inputs.

3 - FUTURE EXTENSIONS

The most useful result of the first implementation of an image understanding vision system is the identification of key problems in the photo interpretation task. These results have guided plans for future extension of the system toward a next-generation vision system.

Beginning at the bottom level of image analysis, it has become clear that more robust means of extracting both intensity edges, shadow boundaries, and region borders are needed. This will be a substantial area of effort in the second contract phase. Region growing techniques, combined with more generalized edge detection, will be explored.

While the modeling and prediction capability of the current implementation will form an important piece of the new vision system, extended capabilities will need to be added to perform prediction of complex shapes generated in situations where object occlusions and detailed shadows cast on nearby 3-D objects must all be considered to identify individual objects. This will include very detailed models, with a more general capability to model irregular shapes, and a projection scheme to predict exact shapes as would be seen in image data. These detailed predictions require exact modeling conditions. The current symbolic prediction system will provide an important coarse pass capability, to establish specific model positions and orientations.

Several issues regarding the vision system performance will be addressed by an intelligent planner system. This system will select models, control coarse pass matching as well as detailed passes, select appropriate low level processing to optimize feature extraction, and select appropriate shape extraction heuristics. Shape extraction will be further enhanced by the direction of shape extraction in an ordered manner, searching first for dominant subpart shapes, and then directing local searches for more detailed subparts and shadow shapes based upon modeled spatial relationships.

At an even higher level of control, an expert system will attempt to emulate some of the procedural expertise applied to the image understanding task by a human photo interpretation specialist. This knowledge will direct application of the vision system to perform specific analysis tasks, such as scene mensuration. It will also determine what results are important, when satisfactory results have been obtained from the vision system, and will be capable of the sort of temporal analysis now captured in the situation analysis module.

4 - CONCLUSIONS

Having concluded Phase One efforts on the Image Understanding program, significant progress has been made toward technological capabilities to automate portions of the photo interpretation task. In its present form, the system is capable of identifying generic classes and subclasses of objects by matching basic dimensions and spatial relationships. Performance is rudimentary in comparison to human capabilities; several areas have been identified for improvement. Individual progress in the areas of shape and shadow prediction, shape extraction, and scene interpretation all have contributed. However, the most significant mark is the inclusion of a complete set of modules for performing object identification in real imagery in a single integrated system, providing a basis for future extension as well as evaluation of other techniques.

5 - ACKNOWLEDGMENTS

The successful completion of the Image Understanding vision system was greatly supported by the advice and efforts of Dr. Rodney Brooks. Both his actual coding of the original Acronym system, as well as much of the extensions for shadow capability and improved interpretation, helped make this a working system. Thanks are due also to Barry Soroka and Tom Binford for their early help in focusing research efforts. Support from Andy Hanson and David Kashton has helped greatly in dealing with the Eunice environment.

REFERENCES

[1] T.O. Binford, "Visual Perception by a Computer," IEEE CONF. ON SYSTEMS AND CONTROLS, Miami (December 1971)

[2] R.A. Brooks, "Goal-Directed Edge Linking and Ribbon Finding," PROCEEDINGS: ARPA IMAGE UNDERSTANDING WORKSHOP, (April 1979)

[3] R.A. Brooks, R. Greiner, and T.O. Binford, "The ACRONYM Model-Based Vision System," PROCEEDINGS: IJCAI-6, Tokyo (August 1979), pp. 105-113

[4] R.A. Brooks, "Symbolic Reasoning Among 3-D Models and 2-D Images," ARTIFICIAL INTELLIGENCE 17, (1981), pp. 285-348

[5] R.A. Brooks, "Symbolic Reasoning Among 3-D Models and 2-D Images," Ph.D. Dissertation, Stanford University, (1981)

[6] R.A. Brooks, ACRONYM MANUAL, unpublished

[7] B.L. Bullock et al, "Image Understanding Application Project: Status Report," PROCEEDINGS: DARPA IMAGE UNDERSTANDING WORKSHOP, (1982)

ROBOT VEHICLES: A SURVEY AND PROPOSED TEST-BED FACILITY

By

Bruce J. Schachter
Glenn E. Tisdale
Westinghouse Electric Corporation
Defense and Electronics Center
P.O. Box 746, MS-440
Baltimore, MD 21203
and
George R. Jones
Army Night Vision and Electro-Optics Lab
Attn: DELNV-AC
Fort Belvoir, VA 22060

ABSTRACT

Many groups are now attempting to build autonomous land robots. However, no vehicle fitting this description exists today. The difficult problems that must be solved are obstacle avoidance and scene matching to map coordinates.

Autonomous robot vehicles would be used primarily by the military for reconnaissance. Enemy targets would either be designated by laser spotting or their locations and identities transmitted to friendly forces.

Westinghouse as a subcontractor to the Computer Vision Lab of the University of Maryland, with the support of the Defense Advanced Research Projects Agency and the U.S. Army Night Vision and Electro-Optics Laboratory, is developing a test-bed facility for investigating autonomous vehicle navigation. A Westinghouse vision system will be added to an existing electronically controllable Army Vehicle.

1. INTRODUCTION

Since April 1976, the Computer Vision Laboratory of the University of Maryland, in collaboration with Westinghouse as a subcontractor, has been engaged in Image Understanding research with the support of the Defense Advanced Research Projects Agency and the U.S. Army Night Vision and Electro-Optics Laboratory. This research has resulted in a number of significant advances in the field of image understanding, with particular emphasis on techniques applicable to tactical imagery. A new research project has started which will build on this work to develop an autonomous vehicle navigation system.

Section 2 discusses the need for autonomous vehicle navigation. The main military application is the detection, recognition, and reporting of rear-echelon enemy targets. This mission can be accomplished by land or air. The unmanned ground vehicle offers some advantages over its airborne counterpart, while allowing penetration into territory generally considered too risky for a man. Section 3 surveys the state-of-the-art in vehicle development. Land, air, and undersea vehicles are all covered since many of the computer, command-and-control, and sensor problems are common to them. Section 4 describes a test-bed facility proposed for implementation at Westinghouse. The test vehicle will have a real-time Westinghouse vision system on board. One goal of this project is to test software developed by other research groups, as well as that originating at Westinghouse and the University of Maryland.

2. THE NEED FOR AUTONOMOUS VEHICLE NAVIGATION

The Army's emerging doctrine of "deep attack" on second and later echelons of the enemy offensive requires that artillery and aircraft strikes be directed against these more distant targets. Successful use of this fire power is dependent upon obtaining accurate target positions and the ability to provide precise weapon delivery. Current developments in fire-and-forget weapons offer the prospect of major advances in weapon delivery precision, once the targets are located. However, initial detection and location of rear-echelon targets remain difficult reconnaissance tasks. This project is directed toward the solution of this problem.

The desired reconnaissance can be accomplished by land or air. The land mission is presently accomplished by "spotters" who are positioned behind enemy lines to locate and report by radio on enemy target concentrations (figure 2-1).

Forward air controllers assist in air strikes which employ laser guided missiles by designating key targets with laser beams. The risks involved in these assignments have spawned various attempts to automate the spotter process. Sensor packages for vehicle detection were developed at the time of the Vietnam conflict for air drop in enemy territory. One of the major problems faced by this Air Force program was the need to establish the exact locations of the sensor packages themselves. Another was the fact that these immobile packages often came to rest in positions from which no detections could be made, or from which their detection by the enemy was easy. In an Army program carried out about the same time, a mobile system was developed for protection of railroad equipment by placing sensors on a small, unmanned railroad car moving at a safe distance ahead of the locomotive. The function of the sensors was to detect any anomalies in the track condition, including placement of mines near the tracks.

The risk suffered by spotters and the various attempts to automate the land reconnaissance activity indicate the value of close-up target information. However, collection by air is an alternate approach. Standoff target acquisition systems, which use long range sensors located above friendly territory, lack the resolution (if not the range) to provide detailed data on rear-echelon targets. Low-level reconnaissance aircraft which penetrate enemy territory are severely restricted by the time available for target search, and by the masking of trees or buildings. Helicopters offer increased search time at the risk of reduced survivability. Remotely piloted vehicles (RPVs) will relieve some of the constraints on search time and survivability, but are not expected to produce information which is competitive with ground reconnaissance. However, the navigation techniques to be developed for this project will apply to aircraft as well as land vehicles.

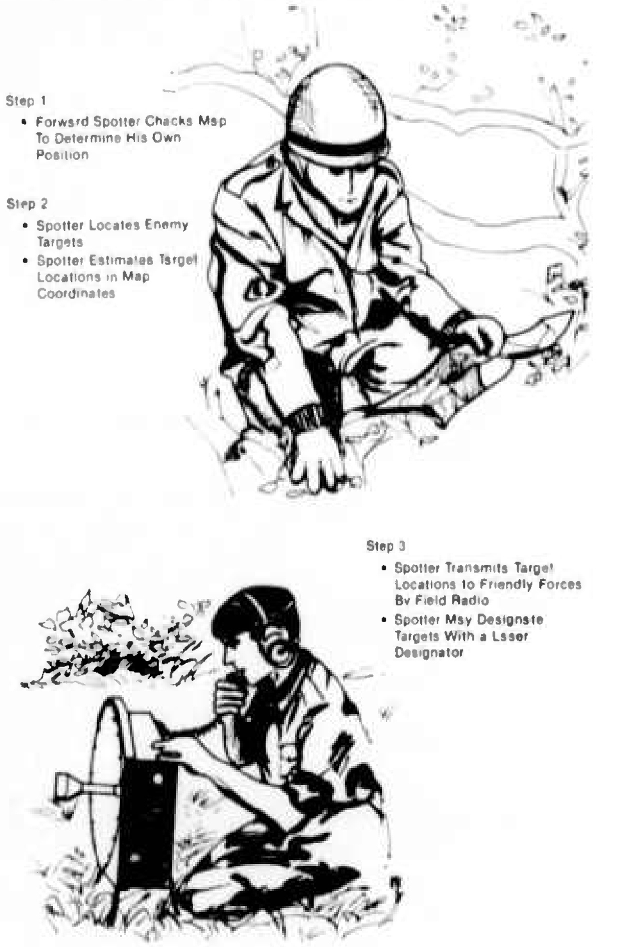

Step 1
• Forwsrd Spotter Chacks Msp To Determine His Own Position
Step 2
• Spotter Locates Enemy Targets
• Spotter Estimates Tsrget Locations in Map Coordinates
Step 3
• Spotter Transmits Target Locations to Friendly Forces By Field Radio
• Spotter Msy Designste Targets With a Lsser Designator

Figure 2-1. Forward Spotter Scenarlo

The above discussion suggests the need for an autonomous land vehicle which can be delivered deep inside enemy territory and which can

- establish its position in map coordinates,
- navigate in a limited way through its environment,
- detect and identify targets and transmit their locations and identities to friendly forces, or designate them by laser spotting.

This concept is illustrated in figure 2-2.

The functions of an autonomous land vehicle, as stated above, represent several areas of advanced research, including scene matching to map coordinates, automatic target recognition, obstacle avoidance, and robotics. All of these areas have been under recent investigation; of the four, however, it is felt that scene matching to map coordinates will require by far the greatest advances in order to establish the feasibility of the land vehicle concept. The robotics area is currently receiving a high level of research support, both military and industrial, which is expected to solve many of the control problems.

Automatic target recognition devices which discriminate between target classes are now in the advanced development stages in the Army and Air Force. Scene matching between similar views of the same area is also a well-established capability. However, the challenge for the land vehicle is to determine its position in map coordinates based upon a series of narrow-field observations taken from a limited series of vantage points.

Another challenge is obstacle avoidance. Large military vehicles have little difficulty traversing ruts and small obstacles, but must avoid barriers, "traps," and cliffs. Most current research on obstacle avoidance centers around the use of laser range data.

2.1 Autonomous Ground Vehicles

As stated above, the ground vehicle should have the ability to perform limited maneuvers so as to improve its vantage point, and avoid immediate detection. It must be capable of detecting and avoiding large obstacles, based upon the interpretation of imagery obtained from its sensors. Infrared, TV, laser-range, and radar sensors, as well as combinations of them, will be considered for use in this project.

The land vehicle possesses several advantages over the airborne platform. It offers long periods of observation at ranges at which detailed target identification can be carried out. Its position can be quite accurately defined. Furthermore, it presents a stable platform for designation.

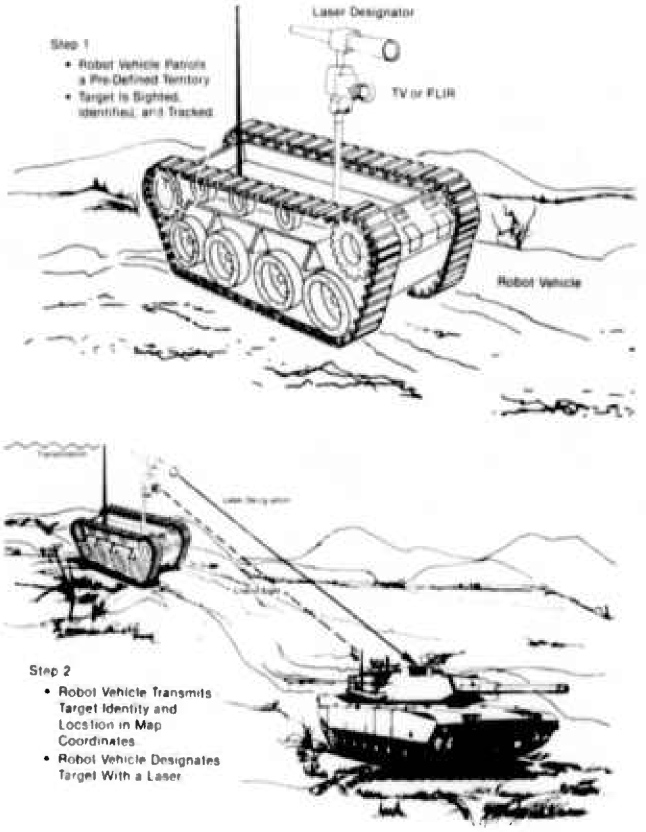

Laser Designator
Step 1
• Robot Vehicle Patrols a Pre-Defined Territory
• Target is Sighted, Identified, and Tracked
TV or FLIR
Robot Vehicle
Step 2
• Robot Vehicle Transmits Target Identity and Locstion in Map Coordinates
• Robot Vehicle Designates Target With a Laser

Figure 2-2. Ground Rohot Vehicle Scenarlo

2.2 Autonomous Alrborne Vehlcles

The Remotely Piloted Vehicle (RPV) offers another solution to the reconnaissance problem (figure 2-3). However, an RPV scenario (figure 2-4) introduces complications of its own. In its current state of development a video link is needed between the RPV and the ground. Imagery is transmitted over this data link, and the image processing tasks of target detection and recognition are performed at a ground control station. The limitations of this approach are its bandwidth requirements and the possibility of the enemy jamming or intercepting the transmission. These limitations can be overcome by reducing the data to be transmitted, encoding it, and having the RPV operate autonomously, using onboard sensor data.

One solution (figure 2-5) is to have an onboard vision system which will:

- navigate the air vehicle through the environment,
- locate, identify, and track enemy targets, and
- transmit target locations and mark them with coded laser spots to direct Hellfire antitank missiles and precision guided ordnance.

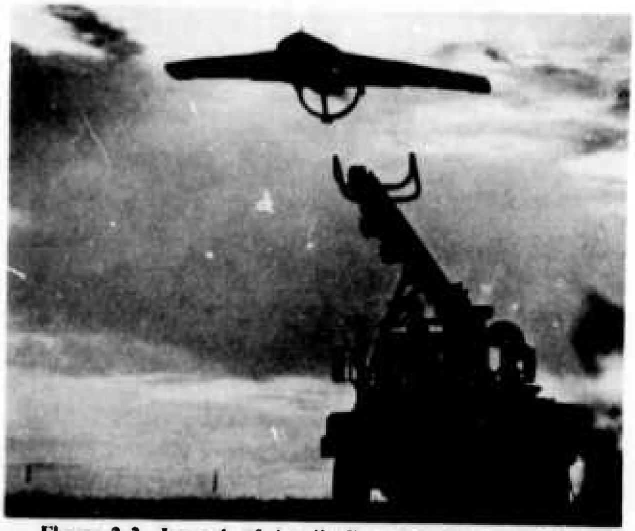


Figure 2-3. Launch of Aquila Remotely Piloted Vehicle

2.3 Application to Current Scenarios

Even though the proposed vision system is intended for use in an unmanned vehicle, it may also find application in a manned vehicle, such as a helicopter. Its function would be to partially or wholly assume certain visual tasks of the pilot, so as to reduce the burden on him during battlefield conditions. In particular, the ability to "hand off" acquired targets to a ground pilot or to another helicopter in map coordinates is greatly desired as an automatic function, but is presently not available.

3. A SURVEY OF AVAILABLE AUTONOMOUS AND REMOTELY CONTROLLED VEHICLES

Despite the impression given to the general public by science fiction movies and the popular press, autonomous outdoor land robots are not now in existence. However, some experimental test-bed facilities are now being built. The state of the art is considerably more advanced for remote-controlled vehicles; air, undersea, and land vehicles are now in production. We will review work in these areas now taking place in the United States. The considerable research and development effort now underway outside the U.S. will also be mentioned.

3.1 Remote-Controlled Vehicles

A large number of remote-controlled vehicles have been built for use in the air, under water, and on land. Much of the technology developed for remotely controlled vehicles can be applied directly to autonomous vehicles.

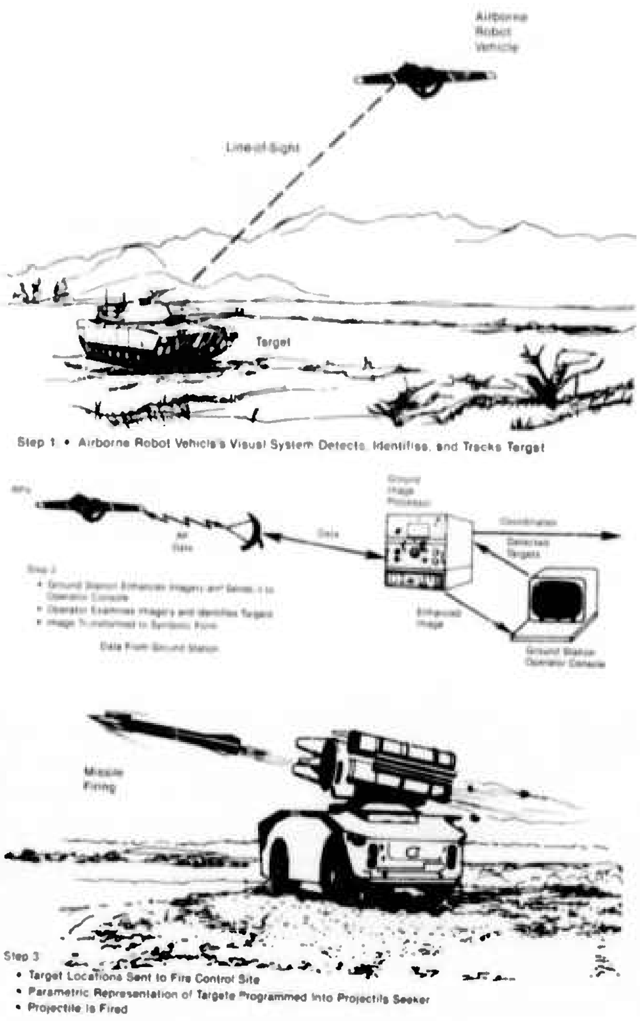

Airborne Robot Vehicle
Line-of-Sight
Target
Step 1 • Airborne Robot Vehicle's Visual System Detects, Identifies, and Tracks Target
RPV
Air Data
Data
Ground Image Processor
Coordinates
Detected Targets
Enhanced Image
Ground Station Operator Console
Step 2
• Ground Station Enhances Imagery and Sends it to Operator Console
• Operator Examines Imagery and Identifies Targets
• Image Transformed to Symbolic Form
Data From Ground Station
Missile Firing
Step 3
• Target Locations Sent to Fire Control Site
• Parametric Representation of Targets Programmed Into Projectile Seeker
• Projectile Is Fired

Figure 2-4. Current RPV Scenario

Air Vehicles

The U.S. Army awarded a contract to Lockheed in August 1979 for the full-scale development of the Aquila Remotely Piloted Vehicle (RPV). Delivered hardware will consist of 22 air vehicles, four ground control stations, three launcher subsystems, three recovery subsystems, and 18 mission payload subsystems. Westinghouse is designing and building the mission payload subsystem, which consists of a stabilized daylight TV sensor, laser rangefinder/designator, stabilized optical path, autotracker, and associated electronics and controls. Once a target is located by slewing the TV camera through controls in the ground station, the camera can be switched to automatic track. The boresight laser can be activated either to provide range to targets or to designate targets for precision guided munitions.

The Aquila operator's console includes a teletype for inserting the planned flight profiles; airspeed, altitude, and heading controls; displays for manual air vehicle control; an X-Y plotter for monitoring aircraft position on a map; and an alphanumeric terminal to display vehicle status. The mission payload console provides a similar interface between the operator and mission payload subsystem. The video sensor can be controlled in azimuth sweep, elevation, three-position zoom, autotrack, and laser aim/fire functions. Both consoles provide real-time video display with instant replay capability.

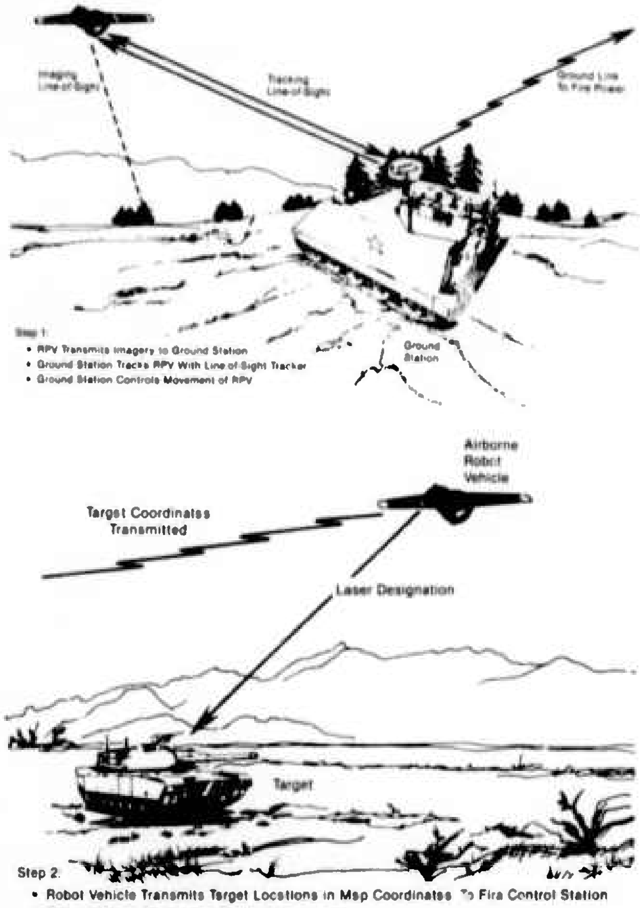

Imaging Line-of-Sight
Tracking Line-of-Sight
Ground Link To Fire Power
Ground Station
Step 1:
• RPV Transmits Imagery to Ground Station
• Ground Station Tracks RPV With Line-of-Sight Tracker
• Ground Station Controls Movement of RPV
Airborne Robot Vehicle
Target Coordinates Transmitted
Laser Designation
Target
Step 2.
• Robot Vehicle Transmits Target Locations in Map Coordinates To Fire Control Station
• Robot Vehicle Designates Target With a Laser

Figure 2-5. Proposed Scenario for Airborne Robot Vehicle

Two RPVs are produced in Israel: the Israel Aircraft Industry Scout and the Tadiran Mastif.[19] They were used for reconnaissance in the destruction of Syrian surface-to-air missile batteries in Lebanon's Bekaa Valley in June 1982. They located the precise positions of SAM sites and relayed the video data to a ground station in real-time. The Israeli RPVs are less costly and contain a less sophisticated payload than their American counterpart.

Land Vehicles

A number of remotely controlled vehicles have been built for duty too hazardous for a human. Vehicles have been built for travel through contaminated environments, such as the Three Mile Island nuclear power plant. In fact, the Atomic Energy Commissions of most industrialized nations have remote-controlled-vehicle programs. The two main U.S. companies building remotely operated vehicles are GCA/PaR Systems and Tracor MBA.

GCA/PaR has built about 15 remotely operated vehicles. We will describe their PaR-1 Manipulator Vehicle (PMV) shown in figure 3-1. The PMV provides a mobile, maneuverable platform for robot arms and TV cameras. The vehicle rides on two wide, flat, neoprene belted tracks. The two rear wheels are powered by separate DC motors. The vehicle is turned by running one of the motors faster than the other, or by driving them in opposite directions. The vehicle contains a double-telescoping, rotating, tube assembly upon which manipulator arms and TV cameras can be mounted. Two highly dexterous manipulator arms are available, with load capacities of 100 and 160 pounds, respectively. TV camera arms are mounted on the same assembly as the manipulator arm so that camera motions are synchronized in vertical movement and rotation with the manipulator shoulder housing. The manipulator can operate a number of tools including saws, shears, fasteners, and torches. It can switch between them via remote control.

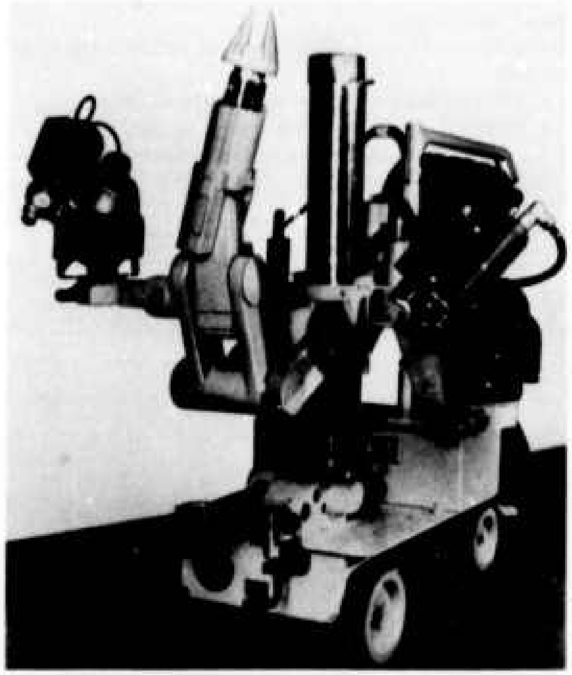


Figure 3-1. PaR Manipulator Vehicle

The PMV control console is portable. It connects to the vehicle by a flexible, quick-disconnect cable. The console provides the necessary control and power for the vehicle and robot arm. It can also include video monitors and camera controls. The following functions of the TV camera can be controlled from the console: lens zoom, iris setting, pan and tilt, and movement of the camera's positioning arms.

Tracor MBA has built about 20 remote-operated vehicles. Their "centipede" has six wheels, TV cameras, and front-mounted manipulator. It is capable of traveling over rough terrain. It can traverse 33-degree slopes, 3-foot vertical barriers, and even climb up an down stairs. The manipulator arms and TV cameras move as slaves to human controlled masters in the vehicle's remote control console. A human controls the arms by an exoskeleton over his own arms. The manipulator arms have force feedback to let the human operator sense the weight of objects remotely handled. The human operator controls the gripping force; it is stated that he can get the remote-controlled arms and hands to perform very delicate tasks such as threading a needle.

The vehicle's two TV cameras can be controlled by a unit mounted on the operator's head. The operator's biocular display provides him with a stereo view. The movement of the slave TV cameras is controlled by the operator's head movement. Control signals and video are transmitted either by cable or by an RF telemetry link. The cable can be up to 300 feet long, the control trailer can be up to 15 miles away if a telemetry link is used.

Other MBA remote-controlled land units include a tracked vehicle, built for the Air Force; an antiarmor vehicle and fork lift, built for the Army; an underground mining vehicle with manipulator, built for the Bureau of Mines; and a driverless tractor remote handling system, built for Lawrence Livermore Lab.

Undersea

Hundreds of remotely controlled underwater vehicles have been built. The reader is referred to the proceedings of the March 1983 conference on remotely operated undersea vehicles.[1]

An example of the state of the art is the Surface Towed Search System developed for the Navy by Westinghouse's Oceanic Division (Annapolis, MD). It is designed to locate and identify objects on the ocean floor at depths of 4 miles.[5]. Currently, computers on board the surface vessel handle signal processing, data recording and analysis, control of the sidelooking sonar, and towing operations. Work is underway at Westinghouse to give more autonomy to submersibles. The emphasis is on the use of onboard expert systems and sonar image processing. Phil Schweizer of Westinghouse, together with Charles Thorpe and Dave McKeown of Carnegie-Mellon University, have developed an algorithm for autonomous vehicle navigation. It matches sets of objects on sonar images with known landmarks.

3.2 Robot Vehicles

Robot vehicles will be described under five headings: (1) legged, (2) indoor and flat-surface, (3) military land units, (4) underwater, and (5) airborne. Although none of the ground units described is fully autonomous, this is the long term goal of their builders.

Legged Vehicles

Robert McGhee of Ohio State University considers legged vehicles particularly appropriate for travel over rough or mucky terrain. He has built a small six-legged walking machine called a hexapod and is now building a much larger unit. In the near future, McGhee believes that some type of human control will be required to guide the vehicle over terrain.[2] One approach is for a human to point to terrain spots with a laser and for the vehicle's sensor system to detect and direct the vehicle to follow the spots. Although in experiments, the hexapod has already been demonstrated in an autonomous mode with onboard computers used for supervisory control, McGhee doesn't see real autonomy coming for several years. He anticipates that autonomous navigation will be based upon active range data used to form a local elevation map, and will not utilize visual scene analysis — at least not in the near future.

McGhee's larger vehicle will have 12 Intel 8086/87 single board computers for motion planning and vehicle control. Specialized hydraulic circuits will be used for drive, lift, and lateral motion. Simple nonscanning proximity detectors may be used for local control of leg motions. Both optical and acoustic sensors are being studied for this purpose.

Odetics has built a large, spindly, radio-controlled, six-legged walking and lifting machine[2] (figure 3-2). It has a clear bubble "head" with built-in TV camera. It is designed to walk over rugged terrain and handle dangerous materials.

Research into walking machines is also taking place at several other locations. Researchers at Moscow University have built at least two hexapods. Marc Raibert of Carnegie-Mellon University has built a one-legged hopping machine designed for studying balance control.[3] Ivan Sutherland of CMU has built a six-legged human-controlled vehicle.[7] Shigeo Hirose of the Tokyo Institute of Engineering has built a four-legged spiderlike machine that can climb stairs.

Indoor and Level-Ground Vehicles

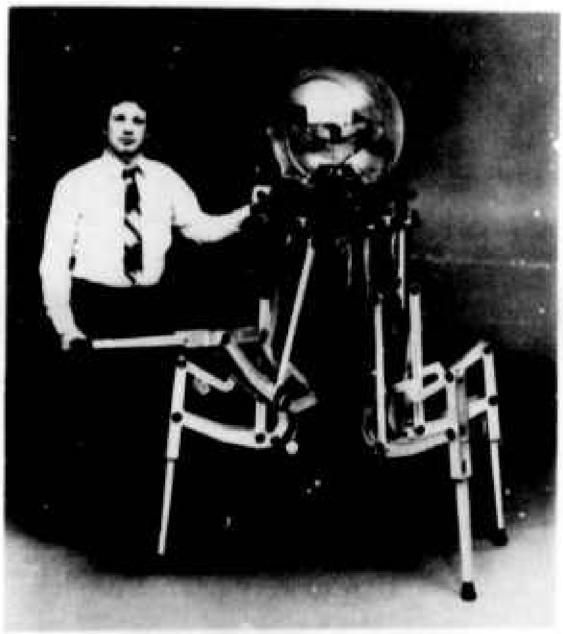


Figure 3-2. Odetics' Odex-1

Hans Moravec of Carnegie-Mellon University has been building robot vehicles with vision systems for a number of years[2] (figure 3-3a). The vehicle now being built (figure 3-3b) will contain 6 to 12 Motorola 68000 single board computers and a TV camera mounted on a pan-and-tilt mechanism. The vehicle runs on three small wheels allowing high mobility over a flat surface. Moravec's extensive experience in this area leads him to believe that the initial step of building and controlling the vehicle is not too difficult to achieve, but it is very hard to get it to do anything significant.

Stanford University has acquired an experimental robot cart from Westinghouse's Unimation Division. The Unimation rover is similar in design and function to the CMU rover. It achieves full three-degree-of-freedom floor-plane mobility on a flat surface.

The World of Robots Corporation is developing a four-wheeled Emergency Security Robot (figure 3-4). Some of its functions have been demonstrated in a remote control mode. The device is intended for automatic patroling, sensing, and acting on intrusion. The unit has five types of sensors: infrared, audio, microwave, ammonia detector, and TV camera. The vehicle has a number of modes of communication: synthesized speech, lights and sirens, and data link. Onboard computers will be used for speech, path following, defense, and reporting.

A number of other robots designed for travel over flat surfaces are listed in table A.

Military Land Vehicles

Military robots can be grouped into two broad classes: static and mobile. A static robot would perform the tasks of point-defense, surveillance, and communication resource management. A mobile robot could be used for mine detection, surveillance; and target detection, recognition, and designation. An intermediate class would consist of vehicles that seek high ground with a clear view, and stop once they find it.

Several groups are trying to robotize existing military vehicles. The interest is in heavy tracked vehicles like tanks and APCs, which can go over small obstacles. Work in this area is being done by Scott Harmon of the Naval Oceans Systems Center, Rosa Chang of FMC Corp., Alexander Meystel of the University of Florida, and several other groups. There is much similarity in the approaches being taken by the separate groups. All vehicles are to be made fully autonomous with onboard expert systems

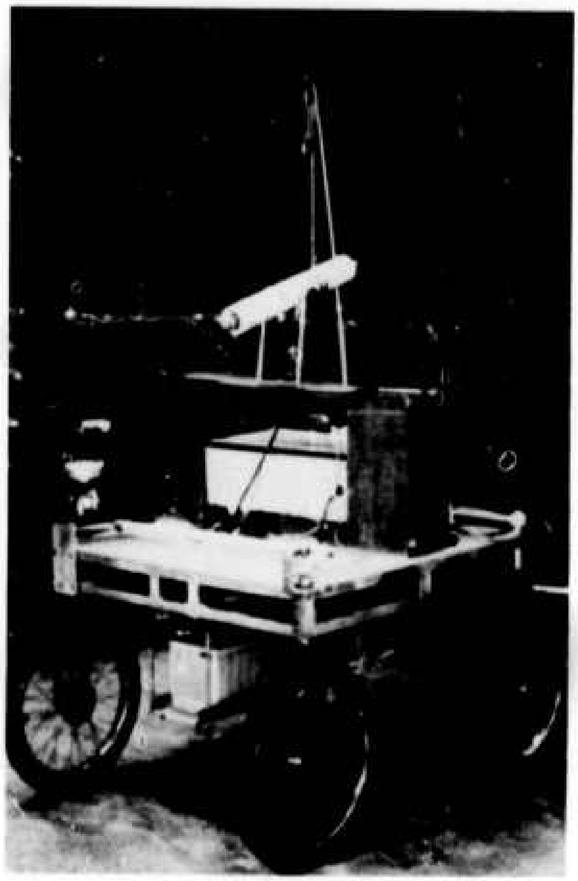


Figure 3-3a. Stanford Cart

and specialized hardware for the processing of visual and/or range images. All groups are using Intel single board computers for general purpose processing. Of the three programs, the one by Scott Harmon is the most ambitious. He expects to have a fully working autonomously navigating vehicle by 1987.

Hughes Research Labs is building a test-bed facility for developing robot vehicle software. They are taking a straight AI approach. They are planning to transition to a real vehicle in 1985-1986, possibly in conjunction with FMC.

The Jet Propulsion Lab vehicle team has suspended work on the Moon and Mars rovers (figure 3-5) and has directed their attention to robotizing military vehicles.

Underwater Vehicles

For a tethered submersible, the drag imposed by the tetherline is a major limiting factor. As the cable length is increased, the cable must be made thicker; the power required to pull it increases accordingly. With four- to seven-mile-long cables, the shipboard cable handling operations become massive and costly. Unmanned, tetherless submersibles are, therefore, particularly desirable for work at great depths. About a dozen vehicles of this type are described in the open literature. The basic characteristics of ten of them are listed in table C.

An autonomous underwater vehicle must be able to determine its own location, locate objects, and perform tasks. It must have command and control software to make decisions and direct operations, since a data link is not possible at these depths. Due to the murkiness of the water, vision is ordinarily possible only at extremely close-in ranges. Increased range can be obtained by imaging sonar systems, however, image resolution and quality are limited.

An example of the state of the art in autonomous submersibles is the University of New Hampshire's EAVE-East vehicle.[9] EAVE-East derives all its information from onboard sensors and is commanded by an onboard expert system. A Motorola SBC is used for command and control. This master computer also directs data acquisition and recording; controls still, movie, and slow-scan TV cameras, and a multibeam sonar system. Three dedicated processors are also on board to perform specific tasks. One controls the thruster speed and vehicle depth, based upon data from pressure sensors. Another is for navigation. It analyzes range and bearing data from remote acoustic transponders. The third dedicated processor handles communication.

Air Vehicles

Existing robot air vehicles are of the "smart missile" type. They are preprogrammed with navigation and targeting information. Various types of active and/or passive sensor data are used during flight. They differ fundamentally from the vehicles described previously since they have a short mission and explode at the end of it.

A cruise missile is basically any unmanned jet aircraft that adjusts its course while traveling to the target.[23] This is accomplished by the use of its inertial guidance system and by the matching of sensed scenes to reference maps. One approach is to compare the sensed terrain contour with digital elevation maps of the flight path.

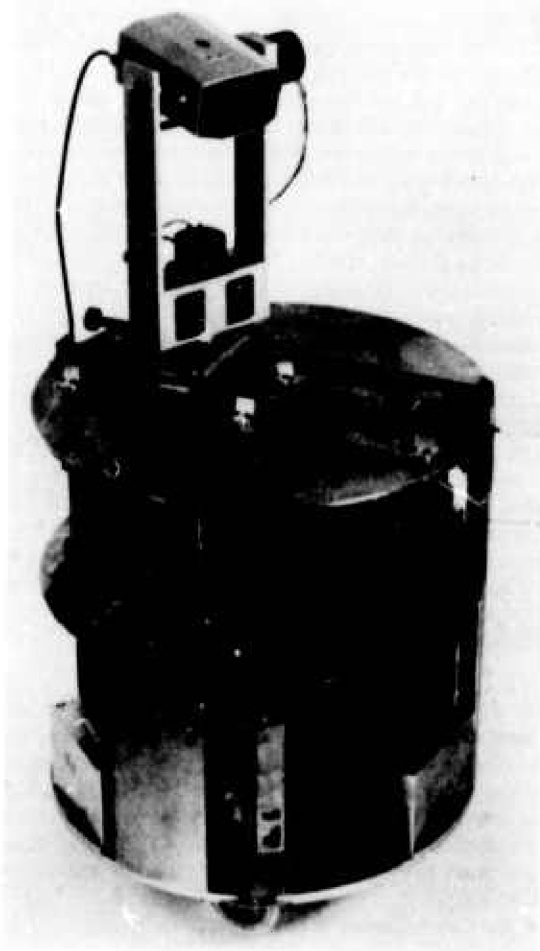


Figure 3-3h. CMU Rover

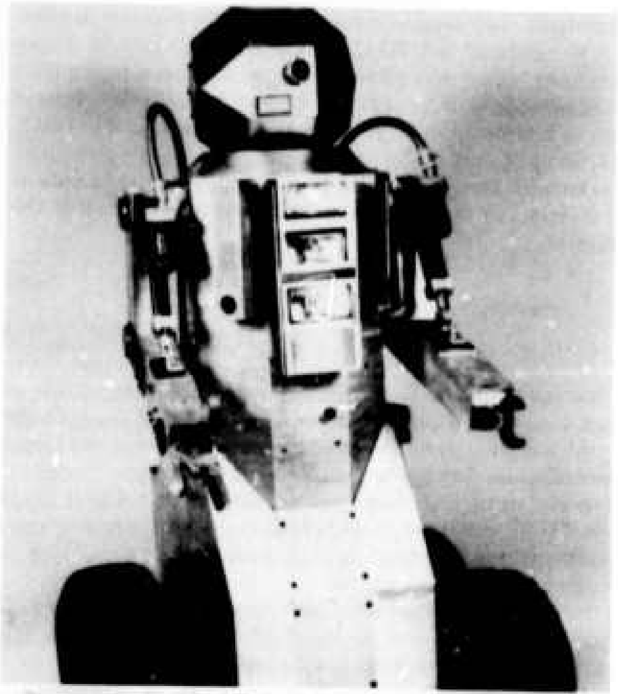


Figure 3-4. World of Robots Emergency Security Robot (Not Fully Operational)

An autonomous tactical missile performs target detection and recognition. Terminal homing is then achieved by the processing of radar, IR, or visible signals.

One example of the state of the art is the French Exocet air-to-sea missile.[20] The aircraft carrying the Exocet flies low over the water beneath the enemy's radar coverage. The pilot then pops the aircraft up to get a bearing on the target, and then drops down again below the enemy's radar coverage. Once the coordinates of the target are determined and passed to the missile, the missile is released in a fire-and-forget mode. The missile heads toward the target primarily under the control of its own inertial guidance system. It uses its radio altimeter to travel just above the water surface. As the missile nears the target it rises slightly, scans the horizon, and locks its terminal homing radar onto the target.

Another type of intelligent weapon system is being developed in the U.S. under the Assault Breaker Program.[22] It works as follows. A missile dispenses "Skeet" delivery vehicles high above an enemy tank concentration. Each delivery vehicle has a parachute that opens at 700 feet to slow its descent. After the chutes are released at 100 feet, the delivery vehicles dispense Skeet submunitions. Each Skeet attempts to locate a tank and fire its self-forging projectile at it.

4. A PROPOSED TEST-BED FACILITY

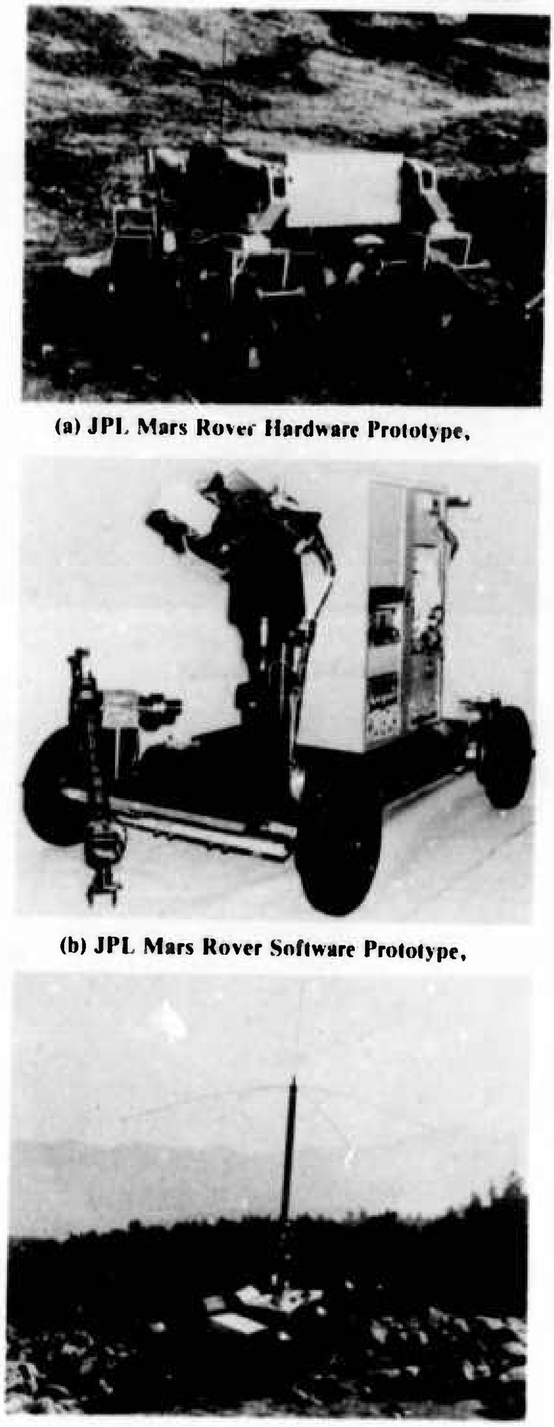
The Army's Night Vision and Electro-Optical Laboratory is providing an Attex all-terrain vehicle (figure 4-1) to the Autonomous Vehicle Navigation Program. This vehicle has been modified by the Systems Integration Division of NV&EOL so that controls can be operated electronically. Westinghouse will add onboard computers, vision system, and controls to convert the vehicle to autonomous operation.

The computer equipment for this project will be configured as two half racks. The two halves will be joined together during laboratory development. The lower half will be disconnected for use on board the vehicle when needed. The upper rack will contain the following equipment:

- monitor
- interactive terminal and keyboard
- Winchester disk drive
- three floppy disk drives.

The lower rack will contain the following equipment:

- Westinghouse real-time gray-level vision system
- Intel chassis, with Intel single board computers (SBCs), core and bubble memories, and interface boards.



(a) JPL Mars Rover Hardware Prototype,

(b) JPL Mars Rover Software Prototype,

(c) JPL Lunar Rover

Figure 3-5. Jet Propulsion Lab Rovers

The vehicle has a telescoping mast upon which will be placed a Fairchild solid-state TV camera and pan-and-tilt mechanism. Other sensors, such as laser range imager, FLIR, acoustic ranger, and radar will be investigated for inclusion. The vehicle will also contain more mundane equipment to make the system work, such as power source and a remote control unit for emergencies and test.

Figure 3-6. Quadractor

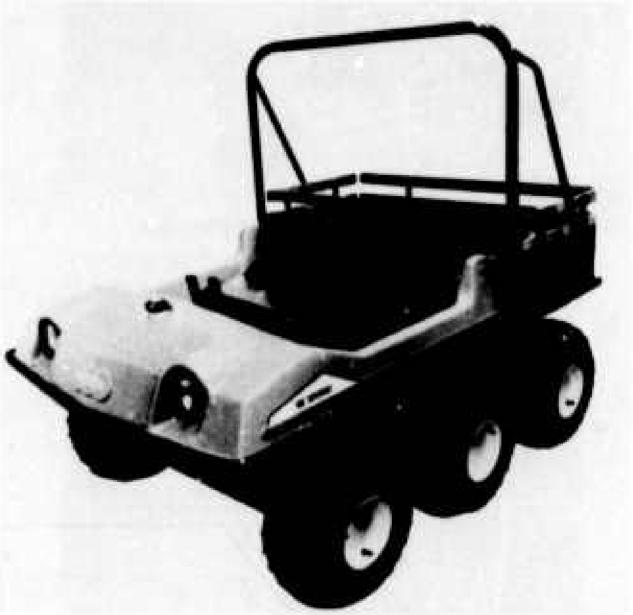


Figure 4-1. Attex All-Terrain Vehicle

The lower rack will be described in more detail. The Intel chassis will initially include two Intel 8086/87 SBCs. Faster, plug compatible SBCs are being developed by Intel each year. They could be substituted or added if required. One-half megabyte of core memory will be present. Five 2-megabyte bubble memory boards will be used for storing the data base. Bubble memory is nonvolatile. Two megabyte boards should be available in 1984.

A Westinghouse AUTO-Q vision system will be used. It performs one billion operations per second to process four million pixels per second. It outputs data of three forms: edge vectors of various lengths, blob data, and area histograms.

Westinghouse is continuing to design new, more powerful vision systems. One unit now in development will produce optical-flow vectors at real-time rates. It may be used later on in this program.

5. CONCLUSIONS

Increased interest is being expressed in the development of truly autonomous mobile robots. The advent of faster single-board computers and denser memories has all but eliminated constraints on processing speed and storage capacity. Dedicated gray-level vision systems are now available to do low-level image processing in real-time. VHSIC hardware will soon be available. Expert systems have been developed which could be adapted to vehicle command and control.

If all this is true, then why aren't robot vehicles ready for full-scale development and deployment? One answer is that the middle-level vision problem has not yet been solved. No software package exists for transforming low-level features into identified objects with known geometric relationships. No one has come even close to solving this problem. It remains to be seen what progress will be made toward the solution in this decade.

Our plan is to construct a state-of-the-art test-bed facility for algorithm evaluation. Since the Intel SBC appears to be emerging as a standard for robot vehicles, it will be possible to evaluate software developed elsewhere as well as software specifically written for this project at the University of Maryland and Westinghouse. Initial research will involve setting simple goals for vehicle navigation within a confined outdoor environment and then trying to achieve these goals. The best hope for progress lies in the cooperation and exchange of results among the various research groups working in this area.

REFERENCES

1. *Proceedings of the Conference on Remotely Operated Vehicles,* San Diego, CA, March 14-17, 1983.

2. H.P. Moravec, *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover,* Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute, Pittsburgh, PA, September 1980.

3. M.H. Raibert, H.B. Brown, M. Chepponis, E. Hastings, S.E. Shreve, and F.C. Wimberly, *Dynamically Stable Legged Locomotion,* Report CMU-RI-TR-81-9, Carnegie-Mellon University, Robotics Institute, Pittsburgh, PA, November 1981.

4. A.S. Westnert, D.R. Blidberg, and R.W. Corell, "Assessment of Technologies in Unmanned, Untethered Autonomous Submersibles," in *Proc of Conf. on Remotely Operated Vehicles,* 1983, pp. 267-271.

5. R.B. Dunbar, "The Role of Tetherless ROVs in Inspection - A Practical Assessment for the Future," *Underwater Systems Design,* Dec. 1982, pp. 9-15.

6. H.P. Moravec, *The Stanford Cart and the CMU Rover,* Carnegie-Mellon University, Robotics Institute, Pittsburgh, PA, February 1983.

7. M.H. Raibert and I.E. Sutherland, "Walking Machines," *Scientific American*, January 1983, pp. 44-53.

8. H.P. Moravec, *Robot Rover Visual Navigation*, UMI Research Press, Ann Arbor, Michigan, 1983.

9. J. Douglas, "Remote Submersibles Take the Plunge," *High Technology*, February 1983, pp. 16-17.

10. H.P. Moravec, "Rover Visual Obstacle Avoidance," *IJCAI 81*, Vancouver, August 24-28, 1981.

11. H.P. Moravec, *Visual Mapping by a Robot Rover*, Stanford University, Artificial Intelligence Lab, April 1979.

12. S.Y. Harmon, *Ground Surveillance Robot*, T.R. 788, Naval Ocean Systems Center, San Diego, CA, September 1982.

13. T. Parrett, "The Appropriate Tractor," *Technology*, January 1982, pp. 57-60.

14. A.D. Carmichael and D.G. Jansson, "Robot II, a Small Unmanned Untethered Underwater Vehicle," Conference Record of Oceans '81, Vol. 1, 1981, p. 109.

15. G.T. Russell and J. Bugge, "Automatic Guidance of an Unmanned Submersible Using a Hierarchical Computer Control Strategy," *Oceans '81 Conference*, IEEE/MTS, September 1981, pp. 118-122.

16. R.B. McGhee, "Vehicular Legged Locomotion," in *Advances in Automation and Robotics*, Ed. by G.N. Saridis, Jai Press, 1983.

17. S. Hirose and Y. Umetani, "The Basic Motion Regulation System for a Quadruped Walking Machine," ASME Paper No. 80-DET-34, *Design Engineering Technical Conference*, Los Angeles, CA, September 1980.

18. S.J. Tsai, *An Experimental Study of a Binocular Vision System for Rough-Terrain Locomotion of a Hexapod Walking Robot*, Ph. D. Dissertation, Ohio State University, Columbus, OH, March 1983.

19. D.M. Russell, "Israeli RPVs: The Proven Weapon System DoD Will Not Buy," *Defense Electronics*, March 1983, pp. 86-94.

20. R.F. Walker, "Smart Weapons in Naval Warfare," *Scientific American*, May 1982, pp. 53-61.

21. *Aviation Week and Space Technology*, Robotic Device May Have Application in Aerospace," April 4, 1983, pp. 59.

22. P.J. Klass, "Microprocessor Controls Antitank Skeet," *Aviation Week and Space Technology*, March 22, 1982, pp. 66-67.

23. F.W. Tortolano, "Unmanned Aircraft Parlay Finesse With Clout," *Design News*, April 11, 1983, pp. 57-66.

Table A. Land Vehicles

NAME	ORGANIZATION	NUMBER BUILT	COST	SIZE WEIGHT PAYLOAD	STATUS	ONBOARO COMPUTERS	POWER	SENSORS. VISION SYSTEM	COMMENTS
Westinghouse Power Plant Vehicle	(W) Hanford, WA Bill Jenkins	1 built 1 bought from PaR and modified	PaR Vehicle cost $30,000 to modify	1 small 1 large with 600 lb payload	Working systems	None	Two ½HP DC motors - AC via trailing cord		Tracked. Umbilical cord. Solid State control system. ac/dc converter.
Rescue Vehicles for Nuclear Power Plants	GCA/PaR Systems Redwing, MINN Will Bochlke 612-484-7261	15	$100,000 minimum	500 lb payload - 1000 lb	Built to order	None	Power via trailing cable	TV cameras	Umbilical cord.. For nuclear industry. Tracked vehicles. Mounted TV cameras.
Mining & Disarming Units	MBA Associates San Remone, CA Marketing Dept 415-837-7201	20	$150,000 and up	Various	Built to order	None	Battery powered & gas powered engines	TV cameras	Steered via radio link. TV camera & video link. Tracked vehicles, some for AF ordnance retrieval.
Odex 1	Odetics, Inc. 1380S Anaheim Blvd., Anaheim CA 92805 Joe Slutzky 714-774-5000	1	$200,000	36" tall, 370 lb payload when stationary, 1000 lb payload when walking	Prototype built	7 micros: Intel 8086, Motorola 68000	24V aircraft battery, 360 W-H	TV cameras	6 legs, 1 micro for each leg. Remote control, r.f. link, joystick. Outdoor vehicle.
RPI Mars Rover	Rensselaer Poly Inst. ECSE Dept Troy, NY 12181 David Gisser 518-270-6485	1		Size of subcompact car, 600 lbs	Not completely operational.	Limited onboard processing plus high frequency data link to mainframe computer.	4 - 6 12V batteries	Triangulating system using solid state pulsed laser, rotating mirror, rotating mast.	Current effort emphasizes vision.
Emergency Security Robot	World of Robots Co., 2335 E High St., Jackson, Mich. 49203 800-248-0896	1	Undetermined	6 ft tall 1000 lbs.	In research and development	Yes	Operates for 12 hrs before batteries need re-recharging	5 sensors: IR, TV, microwave, audio & ammonia. Visual change detection.	4 lg. wheels. Oata link.

Table A. Land Vehicles (Continued)

NAME	ORGANIZATION	NUMBER BUILT	COST	SIZE WEIGHT PAYLOAD	STATUS	ONBOARD COMPUTERS	POWER	SENSORS, VISION SYSTEM	COMMENTS
Robot Army M-114 APC	Naval Oceans Sys. Ctr., Code 8322, San Diego CA 92152 Scott Harmon 619-225-2083	1	Large budget	5 1/2 ft tall, 6 ft wide, 15 ft long, 15000 lbs, several thousand pound payload	Beginning 3 - 4 yr program	20 SBCs Intel 8088 etc., array processor Loosely coupled network PL/M, Pascal		Acoustic sensors, laser range data Vision system planned	Major effort. Point to point autonomous travel planned
Quadractor	Traction, Inc PO Box 90, North Troy, Vt 05859 802-988-4111	300	$3500 for human controlled vehicle $25,000 for electronic steering & braking	64" tall 63" wide 86" long 900 lbs	for sale	None	4 cycle 8 hp Briggs & Stratton gas engine	None	4 tractor wheels, 4 wheel steering 4 wheel drive, flexible frame, powered suspension. Wheels are on leg assemblies (see figure 3-6)
Hexapod	Ohio State Univ Dept of Elect Eng., 2015 Neil Ave., Columbus OH 43210 R. B. McGhee 614-422-2820 also Battelle Labs 505 King Ave Columbus, OH 43201 B. Brownstein, J. Reidy	1 lab model built, 1 being built by 1984	considerable	Current hexapod is small. New one will weigh 9000 lbs	New unit will be completed in 1984 - 1986	New unit: 12 8086 SBCs Pascal		Current unit has 2 GE CID cameras Range sensors. If advanced vision system is added, it will be in 1986 — — — Sensors being investigated by Brian Kelly of Battelle	Current unit is an indoor vehicle. New unit will be an outdoor vehicle. Both have 6 legs — — — Guidance and navigation algorithms under study at Battelle
Heathkit Hero 1 & System Software	TRW, One Space Park, MS 021769 Redondo Beach CA 90278 Mark Thomsen, M Sherbrings 213-535-1706		Low	Based upon Heathkit Hero-1	After hours project at TRW	Yes, see Hero-1 entry		6 Polaroid acoustic range-finders	Emphasis is on software to build a world model
HERO-1	Heathkit 616-982-3411	Many	$2,500	39 lbs.	For sale	Motorola 6808 microprocessor	4 Rechargeable batteries	Sound and light sensor.	3 wheels. Gripper Voice synthesizer A teaching tool
Sumitomo Android	Sumitomo Electric Ind. Osaka Japan	1		36" high 20" wide 39" deep	Prototype built	Yes		2 fiber optic arrays	Gripper. The objective is electronic parts assembly
CMU Rover	Carnegie Mellon Univ., Pittsburgh PA 15213 Raj Reddy Hans Moravec 412-578-3829	1	$50,000 for parts, $250,000 for personnel, $300,000 for systems support. More $ later	60 cm diameter, 100 cm tall, 100 kg	Mechanically complete processors running low-level software partially tested, vision & high level control efforts proceeding in parallel with hardware completion	6 Motorola 68000 SBCs, 10 MC 6805 for servo control language "C"	Batteries & DC-to-DC converter 600 WH, 1200 WH or 2400 WH	TV camera with pan and tilt mechanism. Will be vision controlled	3 degree-of-freedom ground plane mobility. 3 sm wheel assemblies, each able to steer and drive. Data link to VAX 11/780
Wheel Chair Vehicle	Sherry Products Inc., 1501 Pacific Coast Hwy Hermosa Beach, CA 90254 Steve Sherry 213-379-8457	Many	$2000 - $3000	28" wide 39" long 500 lb payload	For sale	None	220 Amp/hr rechargeable 6V battery Twin DC motors, 8 speed auto shift	None	Front wheel steering controlled by magnetic pots. Sturdy outdoor vehicle, can climb hills, has 11" tires. Joystick controlled. Can be easily robotized

Table A. Land Vehicles (Continued)

NAME	ORGANIZATION	NUMBER BUILT	COST	SIZE WEIGHT PAYLOAD	STATUS	ONBOARD COMPUTERS	POWER	SENSORS VISION SYSTEM	COMMENTS
RB5X	R. B. Robot Corp. Suite 201, 14618 W. 6th Ave. Golden, CO 80401		$1495	13" dia. 24" tall 10 lbs.	For sale	Micro-processor with 16 K bytes of core	2 rechargeable 6V gelled electrolyte batteries	Voice recognition. Sonar sensor, bumper switch	RS-232C port. A toy.
Attex All Terrain Vehicle called "Tomahawk"	Vehicle built by Attex Int. Inc. 6168 Woodbine Ave. Ravenna, OH 44266 216-297-0077. Electronic controls added by NV & EOL.	Many human controlled vehicles. 1 robot vehicle		82" long 56" wide 42" high 700 lbs. 700 lb. payload	Electronically controllable vehicle exists at NV&EOL. Three year autonomous navigation program started	Intel 8086 SBCs, core and bubble memory to be added by Westinghouse.	16 HP four cycle gas engine. Batteries.	Sensors and vision system to be added by Westinghouse.	Vehicle has 6 large wheels, telescoping mast. Test-bed facility. U. Maryland is prime contractor.
B.O.B.	Androbots Inc. 1287 Lawrence Station Rd. Sunnyvale, CA 94086. Frank Jones.		$2500	3 ft. tall	For sale	3 microprocessors, up to 3 megabytes of onboard memory.			Toy.
Hughes Testbed	Hughes Research Labs. Malibu, CA 90265. David Tseng. Bruce Bullock.	Several small testbeds	Low	3 ft long 1 1/2 ft wide	Currently testbed for software shakedown. To transition to real vehicle in 1985-1986.	Telemetry link to Lisp processor and vision system.		Vidicon camera (also stored maps). Polaroid sonar sensors. Multiprocessor vision system.	AI algorithm development. Situation assessment, fusion of sensor data, script and plan fixing, knowledge acquisition & representation, goal driven message planning. DMA terrain & culture digital data.
Lunar & Mars Rovers	Cal. Inst. Tech. Jet Propulsion Lab., 4800 Oak Grove Dr. Pasadena, CA 91109 Carl Ruoff 213-354-6101, 4864 Ken Holmes	3	High	Lunar Rover: 6ft long, 18" wide, 16" high, 120 lbs. Mars Rover Software Prototype: 4 ft wide, 7 ft long, 6 ft high, 300 lbs. Mars Rover Hardware Prototype: 6 ft long, 4 ft wide, 3 ft high, 300 lbs.	Lunar and Mars rover missions postponed. Work commencing on autonomous military vehicles (funded by Army).	No rover has onboard computers since program did not advance to flight hardware stage. Onboard electronics does exist.	Lunar Rover: auto batteries. Mars rover SP: Rectified 120V. Mars Rover HP: auto batteries.	Lunar Rover: Remote controlled TV camera. Mars Rover SP: stereo computer vision with correlation tracking computation done in lab computer. Mars Rover HP: No vision system.	Lunar Rover: teleoperated, 3 articulated modules each with 2 wheels. Mars Rover S.P.: controlled via tetherline to lab computers, 4 steerable auto wheels. Mars Rover HP remote control via radio link, 4 tracks (loop wheels).
Robot M113 APC	FMC Corp., 1105 Coleman Ave. San Jose, CA Rosa Chang 408-289-2850	Many human controlled vehicles	Expensive	Large military vehicle.	Ongoing research.	Intel 8086		Vision system to be added in 1984.	Informal agreement with Hughes.
Big Track Testbed	Univ of Florida 137 Larsen, Gainesville, Fl 32611. Alexander Meystel 904-392-4964	1 table-top unit	Low	Very small	Ongoing research.	Sinclair home computer. PDP-11/34. Pascal.		Polaroid range sensors.	Algorithm Development: (1) Planner (2) Navigator (3) Pilot.

Table B. Remotely Piloted Vehicles

NAME	ORGANIZATION	NUMBER BUILT	COST	SIZE WEIGHT PAYLOAD	STATUS	ONBOARD COMPUTERS	PROPULSION	SENSORS	COMMENTS
Aquila	Lockheed airframe. Westinghouse mission payload	22 now proposed	Expensive	6' 10"; 12' 9" wingspan, 220 lbs., 60 lb payload + 25 lbs. in parachute bay	In production	Navigation computer & inertial system	Rear mounted 26 hp engine with push propeller	Stabilized TV camera 3 fields-of-view, autotracker, laser rangefinder laser designator, FLIR to be added	Anti-jam spread spectrum data link very small radar profile
Mastiff	Tadiran Israel Elec. Industries 11 Ben Gurion St., Fival-Shmuel PO Box 648, Tel-Aviv 61006 Israel, 03-713111	Many	Low cost	10' 9"; 14' 2" wingspan, 253 lbs., 66 lb payload	In use	Navigation, ground control auto pilot	Mid mounted 22 hp engine with push propeller	Stabilized gimballed TV camera, panoramic camera for still photography	
Scout	Israel Aircraft Ind., Ben Gurion Intl. Airport, Israel 212-620-4400	Many	Low Cost	12' 1", 11' 9" wingspan, 260 lbs., 40 lb payload	In use	Autotrack GCS manual, autopilot preprogrammed.	Mid mounted 18 hp engine with push propeller	Gyrostabilized gimballed TV camera, with 1:15 zoom and a 2 fields-of-view panoramic camera for still photography	

Table C. Tetherless Unmanned Vehicles

NAME	ORGANIZATION	LENGTH WEIGHT PAYLOAD	MAXIMUM DEPTH	SENSORS	MAXIMUM DEPTH	SENSORS	COMMENTS
Deep Mobile Target	E.M.I. Ltd United Kingdom	3.3 m 236 Kg	366 m	Sonar	366 m	Sonar	For ASW and Training
EAVE East	Univ of New Hampshire Marine Program Marine Program Bldg Durham, NH 03824 Dick Blidberg 603-862-1234	1.5 m 363 Kg 45 Kg payload	50 m	Sonar	50 m	Sonar	Pipeline Instrumentation platform. Will contain an expert system. Two Motorola M68000 SBCs 6100 processor, and bubble memory.
EAVE West	Naval Oceans Systems Center San Diego, CA 92152 Bob Wernli	2.7 m 182 Kg 20 Kg payload	600 m	Still camera, lights.	600 m	Still camera, lights.	
Epaulard	C.N.E.X.O. France	4 m 2900 Kg 40 Kg payload	6000 m	Still camera, lights Sonar	6000 m	Still camera, lights Sonar	Preprogrammed. Surveys mining sites. Brings stereo pairs to surface.
Ocean Space Robot	Mitsui Shipbuilding and Engineering Co Tokyo, Japan	4.8 m 2976 Kg No payload	250 m	Still camera, Sonar.	250 m	Still camera, Sonar.	
PAP 104	Societe 104, Meudan, France	2.7 m 800 Kg 136 Kg payload	150 m	TV camera, lights	150 m	TV camera, lights	Wire guided submersible for sea-bed exploration and the identification of underwater objects.
Robot II	M.I.T. Dept. of Ocean Engineering, Cambridge, MA	2.3 m 110 Kg 11 Kg payload	61 m	Sonar	61 m	Sonar	
Self Propelled Underwater Research Vehicle	Applied Physics Lab Seattle, WA.	3 m 454 Kg 45 Kg payload	1829 m		1829 m		
Unmanned Artic Research Submersible	Applied Physics Lab Seattle, WA.	3 m 408 Kg 23 Kg payload	457 m	Sonar	457 m	Sonar	
UFSS	Naval Research Lab Washington, DC.	6 m	457 m	Very low frequency radio navigation.	457 m	Very low frequency radio navigation.	Long range autonomous vehicle.

SECTION III

TECHNICAL PAPERS
(NOT PRESENTED)

KINEMATICS OF IMAGE FLOWS

Allen M. Waxman

Center for Automation Research
University of Maryland
College Park, MD 20742

ABSTRACT

This study concerns a new formulation and method for solution of the "image flow problem." It is relevant to the maneuvering of a robotic system through an environment containing other moving objects and/or terrain. The two-dimensional image flow is generated by the relative rigid body motion of a smooth, textured object along the line of sight to a monocular camera. By analyzing this evolving image sequence, one hopes to extract the instantaneous motion (described by six degrees of freedom) and local structure (slopes and curvatures) of the object along the line of sight. The formulation relates a new local representation of an image flow to object motion and structure by twelve nonlinear, algebraic equations. The representation parameters, termed "observables", are given by the two components of image velocity, three components of rate-of-strain, spin, and six independent image gradients of rate-of-strain and spin, evaluated at the point on the line of sight. This representation is motivated by the deformation of a finite element of flowing continuum. A method for solving these equations was devised and successfully implemented on a VAX-750 computer. A number of examples were explored revealing two classes of ambiguous scenes (i.e., nonunique solutions are obtained). A sensitivity analysis was also begun in order to estimate noise levels in the representation parameters which still yield acceptable solutions. Estimates of computing time required for this approach to image flow analysis indicate that real-time implementation is not out of the question.

1. INTRODUCTION

This study pertains to the field of real-time dynamic image processing with regard to the maneuvering of a robotic system through an environment containing other moving objects and/or terrain. For a robot to accomplish this task, it may need to determine the three-dimensional structure and relative rigid body motions of these objects, and it must extract this information from a two-dimensional, evolving, monocular image field in real time. That is, the motion in space of a rigid, textured object creates an *image flow* at the camera. The information in an image flow is contained not in the images themselves, but rather in the rate-of-change of the image. It is our aim to *invert the image flow* along a line of sight and thereby determine the motion and local structure of an object under view.

Of course, the image flow problem has its counterpart in the realm of visual perception by man and animals; that is the *optical flow* of Gibson (1966) (also see Marr 1982). However, here we shall concentrate not on the biological issues of perception, but rather on an appropriate representation, mathematical formulation, and solution of the image flow problem. By "appropriate representation" we mean the set of *observables* which describe a local image flow and which lead to a useful formulation of the problem, admitting a rapid and stable solution. But in addition, one must be able to extract these observables from the evolving image in an efficient manner. This last point is often ignored, taking the instantaneous velocity field over the entire image as given. Thus, Prazdny (1980) tried to compute the relative motion and depth map of a set of five points (moving as a rigid body) directly from the two components of image velocity associated with each point (assumed to be given). As expected, this method failed when the points were close to each other for then the image velocities of the different points were all very similar, and computational difficulties associated with round-off errors arose. Therefore, this method could not be used to discern local object structure. Moreover, the availability of accurate velocity measurements for many points in an image is not something one can take for granted. Prazdny's study does, however, point to the importance of image velocity gradients.

Our choice of the "appropriate observables" is motivated by the analogy of a local image flow with the *deformation of a finite element* of flowing fluid. As is well known in continuum mechanics, *the deformation of an infinitesimal element* of flowing continuum may be specified by the velocity-gradient tensor, the symmetric and antisymmetric parts of which have clear geometric interpretations and are termed the "rate-of-strain" and "spin" tensors, respectively. This description is manifest in the Cauchy-Stokes Decomposition Theorem (Aris 1962), and may be applied to an infinitesimal area in an image flow as long as the local object structure is sufficiently smooth. Such considerations were first applied to image flows by Koenderink and van Doorn (1975,1976) who, however, made no attempt to implement them in a computational scheme. Moreover, the image velocity-gradient tensor does not reflect the curvature of an object along the line of sight, nor does it provide a sufficient number of relationships to determine all the unknowns (eight in this case). In order to obtain local object curvature as well as slope, and the relative rigid body motion (which constitutes a total of eleven unknowns), one must consider the six independent gradients of the rate-of-strain and spin as well. That is, the second derivatives of the image velocity field are also necessary. This would provide us with a total of twelve nonlinear, algebraic relations between the six parameters of motion, five structure parameters and the twelve observables, which in principle can be solved. Similar ideas have appeared in a paper by Longuet-Higgins and Prazdny (1980), but they too made no attempt to implement them in order to test the feasibility of their approach. They did, however, outline a method of solution which hinges on the existence of the focus of expansion. Unfortunately, their method does not work for planar surfaces, but in addition it is inapplicable to any object not possessing a relative velocity of approach to (or recession from) the observer, for then a unique vanishing point does not exist. The fact that certain aspects of object curvature are manifest in the second derivatives of image velocity had already been noted by Koenderink and van Doorn (1975), relating the sign of the Gaussian curvature to the gradients in the invariants associated with the rate-of-strain and spin tensors.

The fact that our formulation incorporates the gradients of the rate-of-strain and spin tensors implies that we are indeed analyzing the image flow in a finite neighborhood of the line of sight. Just as the Cauchy-Stokes decomposition describes the deformation of an infinitesimal area in the image, the gradients of this decomposition extend the kinematic analysis to a small but finite area of image. Moreover, the potential for a geometric interpretation of these higher derivatives remains, and this is crucial for extracting the twelve observables from an evolving image element. It is our expectation that the observables we utilize here manifest themselves in the rate-of-deformation of the so-called "zero-crossing curves" of the image intensity distribution (Marr 1982). Preliminary work on idealized curves shows this to be the case.

The approach we have developed, like those described above, is applicable only to smooth structures, i.e., finite slopes and curvatures, and so cannot be used directly near a boundary or cusp on an object. (A multi-resolution implementation on smoothed images should obviate most of these limitations.) Moreover, the whole philosophy of image flows is applicable only to objects with texture or features, for otherwise no image deformation would be observed except near an object's boundaries.

In Section 2, we present a systematic derivation of the twelve kinematic equations relating object structure and motion to our representation of the image flow. The method of solution is developed in (Waxman, 1983) where it is found that the solutions divide into two families, one of which is shown to be non-unique, possessing a two-fold ambiguity. Our method of solving the twelve nonlinear algebraic equations exploits a transformation of the image screen coordinates which aligns one image axis with the direction of zero slope on the object at the point of observation. The required transformation angle is itself a part of the complete solution to the image flow equations. (Waxman, 1983) also presents an alternative (and more rapid) method for obtaining this transformation angle from an additional piece of data in the form of a "radial collision time." This "collision time" represents the distance to the object along the line of sight divided by the relative speed of approach; one could imagine obtaining it from a laser Doppler shift and ranging apparatus. A number of example calculations demonstrating the feasibility and stability of the method are considered in (Waxman, 1983) where it is noted that planar surfaces in motion also reveal a two-fold ambiguity.

2. DERIVATION OF THE "KINEMATIC RELATIONS"

The motion of a rigid body in space may be uniquely specified (in some inertial reference frame) by assigning three independent components of translational velocity and three components of rotational velocity to any point within or on the bounding surface of the object. We shall adopt the point on the surface of the object under view which intersects the line of sight from the observer. As is usually done in image flow studies, the object will be treated as stationary with the relative motion through space ascribed to the observer. The kinematic equations to be derived here relate this rigid body motion and the structure of the object's bounding surface in the neighborhood of the line of sight to our representation of the local image flow.

Following Longuet-Higgins and Prazdny (1980), we adopt the coordinate systems shown in Figure 1. The vertex of perspective projection is located at the origin of a spatial coordinate system (X,Y,Z) whose Z-axis is oriented along the instantaneous line of sight directed at the object. This moving coordinate system has three degrees of translational freedom (V_X,V_Y,V_Z) and three degrees of rotational freedom $(\Omega_X,\Omega_Y,\Omega_Z)$. The two-dimensional image to be analyzed is created by the perspective projection of the object and environment onto a planar screen, oriented normal to the Z-axis and intersecting it at Z=1. The origin of the image coordinate system (x,y) on the screen is located in space at (X,Y,Z)=(0,0,1). Thus, a point P in space, located by position vector R, projects onto the screen as point p as shown in Figure 1.

Due to the motion of the observer, the relative motion of point P in space is $-(\underline{V}+\underline{\Omega}\times R)$. In component form we express P's motion through space as

$$\dot{X} = -V_X - \Omega_Y Z + \Omega_Z Y \ , \qquad (1a)$$

$$\dot{Y} = -V_Y - \Omega_Z X + \Omega_X Z \ , \qquad (1b)$$

$$\dot{Z} = -V_Z - \Omega_X Y + \Omega_Y X \ . \qquad (1c)$$

The projected coordinates of P on the screen (located at Z=1) are given by

$$x = X/Z \quad \text{and} \quad y = Y/Z \qquad (2a,b)$$

The corresponding velocities of the image point P are simply $(v_x,v_y)=(\dot{x},\dot{y})$, obtained by differentiating expressions (2) with respect to time and utilizing relations (1). Hence, the image flow field is given by

$$v_x = \left\{x \frac{V_Z}{Z} - \frac{V_X}{Z}\right\} + \left[xy\Omega_X - (1 + x^2)\ \Omega_Y + y\Omega_Z\right] \ , \qquad (3a)$$

$$v_y = \left\{y \frac{V_Z}{Z} - \frac{V_Y}{Z}\right\} + \left[(1 + y^2)\ \Omega_X - xy\Omega_Y - x\Omega_Z\right] \ . \qquad (3b)$$

We can already see from equations (3) that the image flow reflects neither the absolute distance to points on an object, nor the absolute translational velocities through space; both will be scaled by the distance to the object, say, along the line of sight.

In order to discern the local structure of the object in the neighborhood of the line of sight, we shall perform a kinematic analysis of the image flow in the vicinity of the image origin (x,y)=(0,0). But in doing so, we shall need to take various derivatives of the flow (3) with respect to image coordinates, and this will introduce the slopes and curvatures of the object surface on the line of sight. Thus, to the desired degree of resolution, we must be able to describe the neighborhood of the surface around the line of sight as "smooth," i.e., twice differentiable. Given the locally unique surface $Z=\zeta(X,Y)$, we can describe this neighborhood by a Taylor series about the Z-axis:

$$Z = Z_o + \left(\frac{\partial Z}{\partial X}\right)_o X + \left(\frac{\partial Z}{\partial Y}\right)_o Y + \frac{1}{2}\left(\frac{\partial^2 Z}{\partial X^2}\right)_o X^2 + \frac{1}{2}\left(\frac{\partial^2 Z}{\partial Y^2}\right)_o Y^2$$

$$+ \left(\frac{\partial^2 Z}{\partial X \partial Y}\right)_o XY + \ldots \ . \qquad (4a)$$

We can express this in terms of image coordinates by recalling (2) that X=xZ and Y=yZ, hence the implicit equation $Z=\zeta(xZ,yZ)$. This can be converted locally to an explicit surface relation Z=Z(x,y) possessing its own Taylor series,

$$Z = Z_o + \left(\frac{\partial Z}{\partial x}\right)_o x + \left(\frac{\partial Z}{\partial y}\right)_o y + \frac{1}{2}\left(\frac{\partial^2 Z}{\partial x^2}\right)_o x^2 + \frac{1}{2}\left(\frac{\partial^2 Z}{\partial y^2}\right)_o y^2$$

$$+ \left(\frac{\partial^2 Z}{\partial x \partial y}\right)_o xy + \ldots \qquad (4b)$$

as follows. By replacing X and Y in series (4) according to equations (2) and substituting the whole of (4a) back into the right-hand-side of (4a) wherever Z appears, we find, after collecting terms and comparing with (4b), that

$$\frac{1}{Z_o}\left(\frac{\partial Z}{\partial x}\right)_o = \left(\frac{\partial Z}{\partial X}\right)_o \ , \qquad (5a)$$

$$\frac{1}{Z_o}\left(\frac{\partial Z}{\partial y}\right)_o = \left(\frac{\partial Z}{\partial Y}\right)_o \ , \qquad (5b)$$

$$\frac{1}{Z_o}\left(\frac{\partial^2 Z}{\partial x^2}\right)_o = Z_o\left(\frac{\partial^2 Z}{\partial X^2}\right)_o + 2\left(\frac{\partial Z}{\partial X}\right)_o^2 \ . \qquad (5c)$$

$$\frac{1}{Z_o}\left(\frac{\partial^2 Z}{\partial y^2}\right)_o = Z_o\left(\frac{\partial^2 Z}{\partial Y^2}\right)_o + 2\left(\frac{\partial Z}{\partial Y}\right)_o^2 \ . \qquad (5d)$$

$$\frac{1}{Z_o}\left(\frac{\partial^2 Z}{\partial x \partial y}\right) = Z_o\left(\frac{\partial^2 Z}{\partial X \partial Y}\right)_o + 2\left(\frac{\partial Z}{\partial X}\right)_o\left(\frac{\partial Z}{\partial Y}\right)_o \ . \qquad (5e)$$

Equations (5) relate the dimensionless slopes and curvatures (in units of Z_o) of the surface description in space to those in the image coordinate system. All quantities in (5) are evaluated along the line of sight to the object, hence the subscript zero.

Guided by the Cauchy-Stokes Decomposition Theorem (Aris 1962), we proceed to study the deformation of the image flow (3) by forming the image velocity-gradient tensor, and then decomposing it into a sum of symmetric and antisymmetric parts:

$$\frac{\partial v_i}{\partial \xi_j} = \frac{1}{2}\left(\frac{\partial v_i}{\partial \xi_j} + \frac{\partial v_j}{\partial \xi_i}\right) + \frac{1}{2}\left(\frac{\partial v_i}{\partial \xi_j} - \frac{\partial v_j}{\partial \xi_i}\right) \qquad (6)$$

$$\equiv e_{ij} + \omega_{ij} \ .$$

Here, the subscripts i and j can each take the values 1 and 2, with $(v_1,v_2)=(v_x,v_y)$ and $(\xi_1,\xi_2)=(x,y)$. The individual elements of the rate-of-strain tensor e_{ij}, and the spin tensor ω_{ij}, have geometrical significance in describing the deformation of a differential neighborhood of any image point (x,y), though here we focus on the image origin. The symmetric rate-of-strain tensor e_{ij} has three independent elements: e_{xx}=rate-of-stretch of a differential image line oriented along the x-axis, e_{yy}=rate-of-stretch along the y-axis, $e_{xy}=e_{yx}$= one-half the rate-of-decrease of the angle between two differential segments along the image axes. The antisymmetric spin tensor ω_{ij} has only one independent element, ω=rate-of-rotation of the differential neighborhood of image about the origin. An alternative insight may be gained by considering the eigenvalues and eigenvectors of the rate-of-strain tensor (Aris 1962; Koenderink and van Doorn 1975, 1976); whence, one finds that a circular neighborhood of the point (x,y) dilates according to the sum of the eigenvalues (equivalent to the trace of e_{ij}), undergoes a stretch and compression at constant area according to the difference of the eigenvalues (along mutually perpendicular axes aligned with the eigenvectors of e_{ij}) thereby deforming into an ellipse, and then rotates as a locally rigid area element according to the spin ω. This deformation is superposed on a uniform translation of the infinitesimal neighborhood along with the point (x,y) moving with velocity (v_x, v_y). From equations (3) we obtain the expressions

$$e_{xx} = \frac{V_Z}{Z} + \frac{1}{Z}\frac{\partial Z}{\partial x}\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\} + \left[y\Omega_X - 2x\Omega_Y\right] , \qquad (7a)$$

$$e_{yy} = \frac{V_Z}{Z} + \frac{1}{Z}\frac{\partial Z}{\partial y}\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} + \left[2y\Omega_X - x\Omega_Y\right] . \qquad (7b)$$

$$e_{xy} = \frac{1}{2Z}\frac{\partial Z}{\partial x}\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} + \frac{1}{2Z}\frac{\partial Z}{\partial y}\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\}$$

$$+ \frac{1}{2}\left[x\Omega_X - y\Omega_Y\right] . \qquad (7c)$$

$$\omega = \frac{1}{2Z}\frac{\partial Z}{\partial x}\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} - \frac{1}{2Z}\frac{\partial Z}{\partial y}\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\}$$

$$- \frac{1}{2}\left[x\Omega_X + y\Omega_Y + 2\Omega_Z\right] . \qquad (7d)$$

Relations (3) and (7), evaluated at (x,y)=(0,0), describe the rate of translation, rotation, and deformation of an infinitesimal neighborhood of the image element along the line of sight. Taken together, they constitute six independent relations among eight unknowns (three translation rates scaled by Z_o, three rotation rates, and the two surface slopes). Clearly, we do not have as yet a sufficient number of relations to solve for the number of unknowns encountered so far. This motivates our forming the six independent gradients of the rates-of-strain and spin (or equivalently, the independent second derivatives of v_x and v_y). These gradients represent comparisons of the stretch and rotation rates at neighboring points. But rather than analyzing the infinitesimal neighborhoods of two individual points, these gradients extend our kinematic analysis to a finite neighborhood of a single image point. Utilizing equations (7), we derive the following six independent relations:

$$\frac{\partial e_{xx}}{\partial x} = \left[\frac{\partial^2 \ell n Z}{\partial x^2} - \left(\frac{\partial \ell n Z}{\partial x}\right)^2\right]\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\} - 2\frac{\partial \ell n Z}{\partial x}\frac{V_Z}{Z} - 2\Omega_Y , \qquad (8a)$$

$$\frac{\partial e_{xx}}{\partial y} = \left[\frac{\partial^2 \ell n Z}{\partial x \partial y} - \frac{\partial \ell n Z}{\partial x}\frac{\partial \ell n Z}{\partial y}\right]\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\} - \frac{\partial \ell n Z}{\partial y}\frac{V_Z}{Z} + \Omega_X , \qquad (8b)$$

$$\frac{\partial e_{yy}}{\partial x} = \left[\frac{\partial^2 \ell n Z}{\partial x \partial y} - \frac{\partial \ell n Z}{\partial x}\frac{\partial \ell n Z}{\partial y}\right]\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} - \frac{\partial \ell n Z}{\partial x}\frac{V_Z}{Z} - \Omega_Y , \qquad (8c)$$

$$\frac{\partial e_{yy}}{\partial y} = \left[\frac{\partial^2 \ell n Z}{\partial y^2} - \left(\frac{\partial \ell n Z}{\partial y}\right)^2\right]\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} - 2\frac{\partial \ell n Z}{\partial y}\frac{V_Z}{Z} + 2\Omega_X . \qquad (8d)$$

$$\frac{\partial \omega}{\partial x} = \frac{1}{2}\left[\frac{\partial^2 \ell n Z}{\partial x^2} - \left(\frac{\partial \ell n Z}{\partial x}\right)^2\right]\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} + \frac{1}{2}\frac{\partial \ell n Z}{\partial y}\frac{V_Z}{Z}$$

$$- \frac{1}{2}\left[\frac{\partial^2 \ell n Z}{\partial x \partial y} - \frac{\partial \ell n Z}{\partial x}\frac{\partial \ell n Z}{\partial y}\right]\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\} - \frac{1}{2}\Omega_X , \qquad (8e)$$

$$\frac{\partial \omega}{\partial y} = -\frac{1}{2}\left[\frac{\partial^2 \ell n Z}{\partial y^2} - \left(\frac{\partial \ell n Z}{\partial y}\right)^2\right]\left\{\frac{V_X}{Z} - x\frac{V_Z}{Z}\right\} - \frac{1}{2}\frac{\partial \ell n Z}{\partial x}\frac{V_Z}{Z}$$

$$+ \frac{1}{2}\left[\frac{\partial^2 \ell n Z}{\partial x \partial y} - \frac{\partial \ell n Z}{\partial x}\frac{\partial \ell n Z}{\partial y}\right]\left\{\frac{V_Y}{Z} - y\frac{V_Z}{Z}\right\} - \frac{1}{2}\Omega_Y . \qquad (8f)$$

Now if we evaluate equations (3), (7) and (8) at (x,y)=(0,0), there result twelve independent relations between the image flow representation and the eleven parameters describing the rigid body motion in space and object structure in a finite neighborhood of the line of sight. In order to simplify the notation throughout the remainder of the paper, we define the twelve observables O_i (i=1,12) as the following kinematic quantities evaluated on the line of sight (x,y)=(0,0):

$$O_1 = v_x , \quad O_2 = v_y , \qquad (9a, b)$$

$$O_3 = e_{xx} , \quad O_4 = e_{yy} , \quad O_5 = e_{xy} , \quad O_6 = \omega , \qquad (9c\text{-}f)$$

$$O_7 = \frac{\partial e_{xx}}{\partial x} , \quad O_8 = \frac{\partial e_{xx}}{\partial y} , \quad O_9 = \frac{\partial e_{yy}}{\partial x} , \quad O_{10} = \frac{\partial e_{yy}}{\partial y} , \qquad (9g\text{-}j)$$

$$O_{11} = \frac{\partial \omega}{\partial x} , \quad O_{12} = \frac{\partial \omega}{\partial y} \qquad (9k, \ell)$$

Just as the Cauchy-Stokes Decomposition Theorem describes the rate-of-deformation of an infinitesimal area in the image, our observables describe the deformation of a finite area in the image. We refer to them as observables for they constitute our representation of the local image flow, and must be extracted from the evolving image. In future work we expect to relate these observables to the rate-of-deformation of the "zero-crossing curves" of an image which reflect the texture of the object's surface (Marr 1982).

Next, define the six parameters of motion M_j (j=1,6) describing the observer's rigid body motion through space:

$$M_1 = \frac{V_X}{Z_o} , \quad M_2 = \frac{V_Y}{Z_o} , \quad M_3 = \frac{V_Z}{Z_o} , \qquad (10a\text{-}c)$$

$$M_4 = \Omega_X , \quad M_5 = \Omega_Y , \quad M_6 = \Omega_Z . \qquad (10d\text{-}f)$$

The parameters M_1 and M_2 give the angular velocities of the object perceived by the observer due to his translation through space. Parameter M_3 is the inverse of a radial collision time between object and observer due to the relative velocity of approach, when taken alone. Parameters M_4, M_5, and M_6 give the components of object spin perceived by the observer due to his own spinning motion. Similarly, we define five parameters of structure (or topography) T_k $(k=1,5)$, utilizing relations (5), as follows:

$$T_1 = \left(\frac{\partial \ell n Z}{\partial x}\right)_o = \left(\frac{\partial Z}{\partial X}\right)_o \quad . \tag{11a}$$

$$T_2 = \left(\frac{\partial \ell n Z}{\partial y}\right)_o = \left(\frac{\partial Z}{\partial Y}\right)_o \quad . \tag{11b}$$

$$T_3 = \left(\frac{\partial^2 \ell n Z}{\partial x^2}\right)_o - T_1^2 = Z_o \left(\frac{\partial^2 Z}{\partial X^2}\right)_o \quad , \tag{11c}$$

$$T_4 = \left(\frac{\partial^2 \ell n Z}{\partial y^2}\right)_o - T_2^2 = Z_o \left(\frac{\partial^2 Z}{\partial Y^2}\right)_o \quad , \tag{11d}$$

$$T_5 = \left(\frac{\partial^2 \ell n Z}{\partial x \partial y}\right)_o - T_1 \, T_2 = Z_o \left(\frac{\partial^2 Z}{\partial X \partial Y}\right)_o \quad . \tag{11e}$$

Parameters T_1 and T_2 describe the slope of the object surface at the point on the line of sight, and also yield the local elements of the surface metric tensor. Parameters T_3, T_4, and T_5 yield (along with T_1 and T_2) the three independent elements of the (dimensionless) symmetric curvature tensor deseribing the variation of surface slope in the neighborhood of the line of sight (McConnell 1957). The slopes themselves are dimensionless; the curvatures are scaled by the distance Z_o to the object along the line of sight.

Thus, evaluating equations (3), (7) and (8) at $(x,y)=(0,0)$ and incorporating the definitions (9), (10) and (11), we obtain the following twelve kinematic relations describing the image flow in a finite neighborhood of the line of sight:

$$O_1 = -M_1 - M_5 \quad , \tag{12a}$$

$$O_2 = -M_2 + M_4 \quad , \tag{12b}$$

$$O_3 = M_3 + M_1 T_1 \quad , \tag{12c}$$

$$O_4 = M_3 + M_2 T_2 \quad , \tag{12d}$$

$$O_5 = \tfrac{1}{2} (M_2 T_1 + M_1 T_2) \quad , \tag{12e}$$

$$O_6 = -M_6 + \tfrac{1}{2} (M_2 T_1 - M_1 T_2) \quad , \tag{12f}$$

$$O_7 = -2 \, (M_5 + M_3 T_1) + M_1 T_3 \quad , \tag{12g}$$

$$O_8 = M_4 - M_3 T_2 + M_1 T_5 \quad , \tag{12h}$$

$$O_9 = -M_5 - M_3 T_1 + M_2 T_5 \quad . \tag{12i}$$

$$O_{10} = 2 \, (M_4 - M_3 T_2) + M_2 T_4 \quad . \tag{12j}$$

$$O_{11} = \tfrac{1}{2} (-M_4 + M_3 T_2 + M_2 T_3 - M_1 T_5) \quad , \tag{12k}$$

$$O_{12} = \tfrac{1}{2} (-M_5 - M_3 T_1 - M_1 T_4 + M_2 T_5) \quad . \tag{12ℓ}$$

These image flow equations form a set of twelve coupled, nonlinear, algebraic equations among eleven unknowns. The fact that they are nonlinear implies the possibility of multiple solutions, corresponding to ambiguous scenes. In the next section we shall solve these equations, selecting eleven of them in order to recover possible sets of M_j and T_k corresponding to given O_i, and reserving the twelfth equation as a constraint relation which any acceptable solution must satisfy. As we shall see, two classes of ambiguous scenes emerge, but one must keep in mind that these ambiguities are local; a patching together of local solutions to form global structure and motion models may well break these ambiguities in most cases.

Finally, note that the nonlinearities inherent in the image flow equations (12) are all quadratic, being formed by the product of a structure parameter T_k with one of the translational motion parameters M_i $(i=1,3)$. In fact, the slopes T_1 and T_2 are always multiplied by M_1, M_2, or M_3; the curvatures T_3, T_4, and T_5 always appear in products with M_1 or M_2 alone. That is to say, surface slope is revealed by translation through space, while surface curvature is revealed by translation parallel to the image plane. (Recall, however, that the rigid body motion of interest was defined with respect to the point on the object's surface intersected by the line of sight, which is not the dynamical center of mass of the object under view.)

3. SOLUTION BY TRANSFORMATION

By "solving the image flow equations" we mean, given the twelve observables O_i, obtain all possible sets $\{M_j, T_k\}$ of motion and structure parameters that are consistent with relations (12). That is, we wish to invert the local image flow. The difficulty in solving equations (2) stems from the multitude of quadratic nonlinearities formed by products of motion and structure parameters. Moreover, all these parameters may range between plus and minus infinity, in principle. The method of solution developed in (Waxman, 1983) hinges on a rotation of the image coordinate system which aligns one image axis with the direction of zero slope on the object at the point of observation. The transformation angle is itself an unknown which is to be solved for, replacing one of the structure parameters. However, this angle α is bounded, $-90° < \alpha \leq +90°$, and this fact makes all the difference!

Consider the differential of radial distance between observer and object, in the neighborhood of the line of sight. To first order in differential quantities, this is equivalent to the differential dZ evaluated at $(x,y)=(0,0)$. We have

$$dZ_o = \left(\frac{\partial Z}{\partial x}\right)_o dx + \left(\frac{\partial Z}{\partial y}\right)_o dy = Z_o \, (T_1 \, dx + T_2 \, dy); \tag{13a}$$

hence

$$dZ_o = 0 \text{ along } \frac{dy}{dx} = -\frac{T_1}{T_2} \quad . \qquad (13b)$$

Relations (13b) indicate a direction in the image plane, passing through its origin, along which the distance to the object is locally constant, i.e., it is the direction of vanishing slope. (This direction corresponds to an equi-depth contour as is found in Moiré photography (Meadows, et al. 1970; Takasaki 1970) of objects.) This suggests constructing a new set of image coordinates $(\bar{x},\bar{y})$ rotated by an angle α from the original image coordinates (x,y), as shown in Figure 1. By aligning the $\bar{x}$-axis with the direction of zero slope we have, by definition, $\bar{T}_1 \equiv 0$. According to (13b), the transformation angle is uniquely specified by

$$\alpha = \tan^{-1}(-T_1/T_2) \text{ with } -90° < \alpha \leq +90° \quad , \qquad (13c)$$

except at "specular points" where $T_1 = T_2 = 0$. Specular points correspond to local maxima, minima, and saddle points in the distance between observer and object surface. At these points, the differential in distance vanishes to first order in all directions in the image plane; thus, any convenient value may be chosen for α when the line of sight intersects such a point. Of course the rotation angle α, as given by (13c), is not known ahead of time since T_1 and T_2 are themselves unknowns to be solved for. That is, the angle α replaces the first structure parameter (in the transformed system) as an unknown.

Just as we can specify a rotation of the image coordinates for any angle α, we can construct corresponding transformation relations for the observables, the parameters of motion, and the structure parameters, i.e. $\{O_i, M_j, T_k\} \overset{\alpha}{\rightarrow} \{\bar{O}_i, \bar{M}_j, \bar{T}_k\}$. We are, however, interested only in those α for which $\bar{T}_1 \equiv 0$. The required transformations are derived in (Waxman, 1983). The transformed image flow equations, relating $\bar{M}_j$ and $\bar{T}_k$ to $\bar{O}_i$, are simply equations (12) without the terms involving $\bar{T}_1$. The problem is then: given the O_i, find all solution sets $\{\alpha, \bar{M}_j, \bar{T}_k\}$ of the transformed image flow equations and then transform the motion and structure parameters back to the original coordinates $\{\bar{M}_j, \bar{T}_k\} \overset{\alpha}{\rightarrow} \{M_j, T_k\}$.

Solving equations (12) in the transformed system is quite straightforward; and finding the angle α is not difficult either. Solutions generally divide themselves into two classes, one with unique solutions when the surface possesses local curvature, the other being inherently nonunique with a two-fold ambiguity. These ambiguous solutions arise when a particular coincidence between object motion and local structure occurs. "Specular points" are an important exception, giving rise to unique interpretations despite their membership in the second class of solutions. Planar surfaces in motion also possess a two-fold ambiguity, except when there is no relative approach velocity to the observer, in which case the interpretation is unique. However, it should be kept in mind that these multiple interpretations are "local ambiguities," and that many of them may be resolved in the process of building global structure models by piecing together solutions from neighboring regions. A sensitivity analysis was also begun in order to ascertain the effects of noise (or uncertainty) in the observables. Preliminary results suggested that the method is quite stable, though further studies remain to be done in this area (see (Waxman, 1983) for details).

4. CONCLUDING REMARKS

We have introduced a new representation of a local image flow in terms of the image velocities, strain rates, spin, and image gradients of strain rate and spin, evaluated along the line of sight to a moving surface. A set of twelve kinematic relations (nonlinear algebraic equations) were derived which relate these representations parameters ("observables") to the local surface slopes, curvatures, and parameters of rigid body motion. A method to solve these equations for the structure and motion parameters, given the observables, has been developed and implemented on a VAX-750 computer.

The next phase of this work will be directed at extracting the "observables" from an evolving image sequence. In this regard we hope to exploit the neighborhood interpretation of the local image flow representation adopted here. That is, our "observables" actually describe the rate-of-deformation of a small but finite neighborhood in the image, around the line of sight. As the deformation of a neighborhood can be ascertained by studying the deformation of its bounding curve, we expect to be able to obtain all of the observables by following the deformation of closed "zero-crossing curves" of the image intensity variation map. We anticipate that this neighborhood approach should also lead to a rather robust method of obtaining the required observables, more so than tracking a set of points through an evolving image sequence.

ACKNOWLEDGMENTS

The theoretical aspects of this research were initiated by myself, in collaboration with Prof. Shimon Ullman, while both of us were present at the Weizmann Institute of Science, Department of Applied Mathematics (Rehovot, Israel) during the Spring of 1981. It was Prof. Ullman who first brought the image flow problem to my attention, feeling that a hydrodynamicist may be able to shed some light on this matter; and I thank him for that. During the period October 1982 through April 1983, this work was supported by the National Science Foundation ("Small Business Innovation Research" Program) under award No. ECS-8260447 to HYDRONAUTICS, Incorporated where the author was employed as a Research Scientist.

REFERENCES

Aris, R. 1962, "Vectors, Tensors, and the Basic Equations of Fluid Mechanics " (Englewood Cliffs: Prentice-Hall).

DEDUCING FACTS ABOUT SCENES FROM IMAGES

Jussi Ketonen
Stanford University

1. Abstract

This note describes ways of formally expressing facts about images. It is shown that some of the known heuristics for deducing facts about scenes from images can actually be proved correct.

2. Introduction

Our intent is to provide a "mathematical" commentary for some of the recent work on interpretations of image data. In particular, we can show that a few of the coincidence assumptions stated by Binford in [1] can actually be proved in a suitable formal framework.

One should not expect formalisations of theories to have tangible connections with succesfull implementations of algorithms; Artificial Intelligence programs need not be based on the paradigm of theorem proving. However, the clarification of the formal concepts underlying these systems can be of great importance in terms of program architecture and further development. Axiomatization of knowledge domains can therefore be viewed as a pedagogical (or philosophical) exercise, the basic intent being the elucidation of the fundamental concepts involved and their relationships.

We will provide a formal framework for discussing the representation of polyhedra by line drawings. It follows from our analysis that many of the "impossible" pictures of Huffman in [3] can be detected by simpler means than the ones used by Huffman [3], Clowes [2], Waltz [7] or Wesley and Markowsky [4]. Given that our methods are simpler (even if not complete), they may be closer to the process actually used by the human visual system. They have the additional advantage of being generalizable with suitable modifications to semi-algebraic sets (i.e. objects defined through sets of algebraic inequalities), though this will not be taken up in this note. For the sake of simplicity, only objects defined by linear varieties are discussed.

3. Formal Background

We follow the standard conventions of geometry and topology in our notation and terminology. For background, one may consult Rourke and Sanderson [5] or Mumford [6] - the techniques presented in this paper (though mathematically rather trivial) have the flavor of algebraic geometry.

The terminology of Wesley and Markowsky [4] is used to describe the fundamental objects of study.

DEFINITION 1: A *face* f is the closure of a nonempty, bounded, connected, coplanar, open subset of $\mathfrak{R}^3$ whose boundary is the union of a finite number of line segments.

DEFINITION 2: An *object* O is the closure of a nonempty, bounded, open subset of $\mathfrak{R}^3$ whose boundary is the union of a finite number of faces.

From now on, by a *face of an object* O we mean a maximal face contained in ∂O. It is easy to see that any object is a compact polyhedron; in particular a finite union of simplices

DEFINITION 3: The *vertices* of f is the set of all points for which two noncollinear line segments contained in the boundary of f can be found whose intersection is the given point.

DEFINITION 4: The *edges* of f is the set of all line segments e contained in the boundary of f such that all the endpoints of e are vertices and no interior point of e is a vertex.

The vision task may be modeled by a (non-trivial linear) projection $\Pi : \mathfrak{R}^3 \rightarrow \mathfrak{R}^2$. We hypothesize an object O contained in the positive part of $\mathfrak{R}^3$ in general position with respect to Π; i.e., any linear variety generated by the vertices of O is in general position with respect to Π. There is a natural ordering $\prec$ on the fibers $\Pi^{-1}(\{x\})$. Since O is compact, the set $O \cap \Pi^{-1}(\{x\})$, if non-empty, has a $\prec$- least element, which we we denote by S_x.

DEFINITION 5: The set $S = \{S_x | x \in \Pi(O)\}$ is the *visible part* of O under Π.

It follows that $\overline{S}$ is a compact polyhedron and that the map $\Pi : S \rightarrow \Pi(O)$ is bijective. In fact, one can show that for any x

$$|\Pi^{-1}(\{x\}) \cap \overline{S}| < \infty.$$

Gibson, J. J. 1966, "The Senses Considered as Perceptual Systems " (Boston: Houghton-Mifflin).

Koenderink, J. J. and van Doorn, A. J. 1975, "Invariant Properties of the Motion Parallax Field Due to the Movement of Rigid Bodies Relative to an Observer," Optica Acta 22, 773.

Koenderink, J. J. and van Doorn , A. J. 1976, "Local Structure of Movement Parallax of the Plane," J. Optical Soc. America 66, 717.

Longuet-Higgins, H. C. and Prazdny, K. 1980, "The Interpretation of a Moving Retinal Image," Proc. Royal Soc. London B208, 385.

Marr, D. 1982, "Vision: A Computational Investigation into the Human Representation and Processing of Visual Information" (San Francisco: W. H. Freeman).

McConnell, A. J. 1957, "Applications of Tensor Analysis" (New York: Dover).

Meadows, D. M., Johnson, W. O., and Allen, J. B. 1970, "Generation of Surface Contours by Moiré Patterns," Applied Optics 9, 942.

Prazdny, K. 1980, "Egomotion and Relative Depth Map from Optical Flow," Biological Cybernetics 36, 87.

Takasaki, H. 1970, "Moiré Topography," Applied Optics 9, 1467.

Waxman, A. 1983, "Kinematics of Image Flow with Applications to Computer Vision," Technical Report 83018-1, HYDRONAUTICS, Inc.

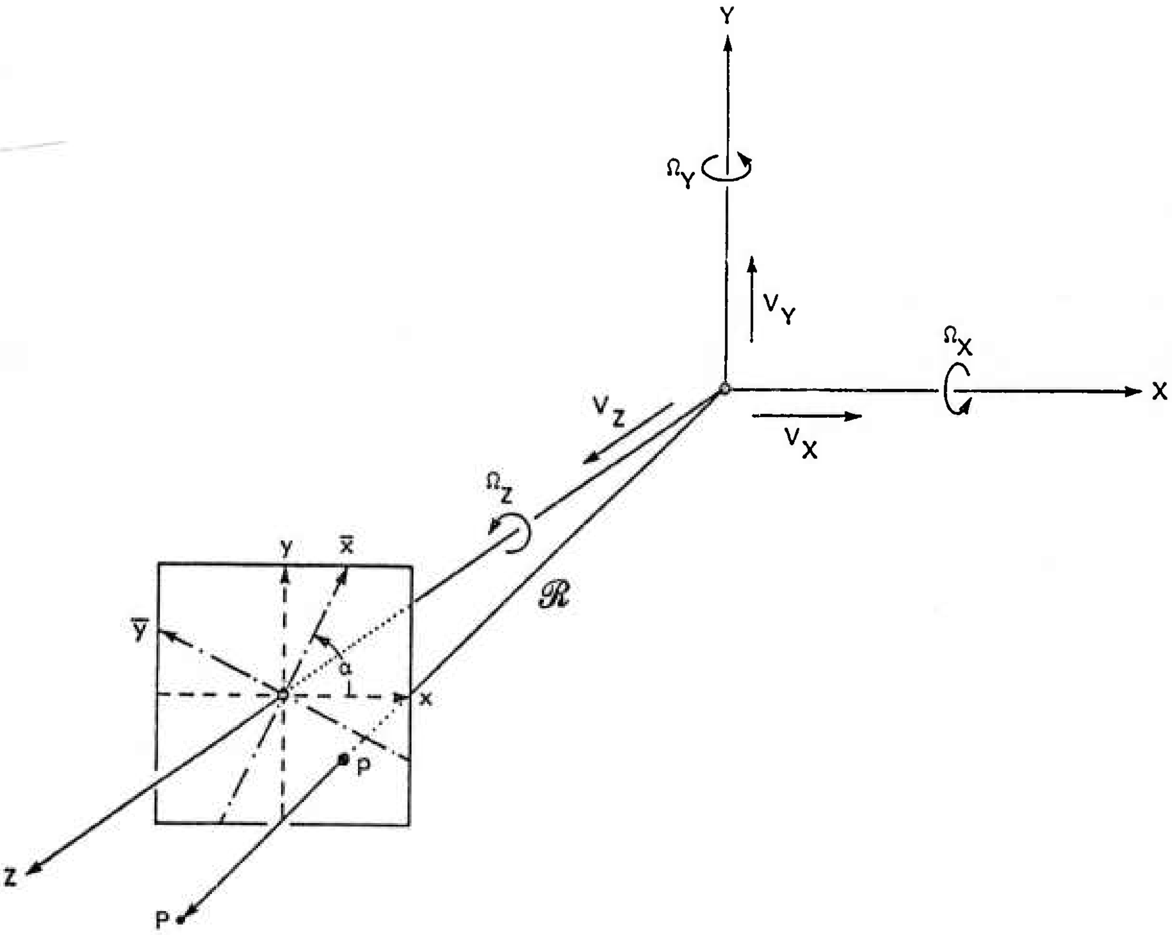


Figure 1. Spatial coordinates moving with the observer, and image coordinate systems.

FRACTAL-BASED DESCRIPTION OF NATURAL SCENES

Alex Pentland
Artificial Intelligence Center
SRI International
333 Ravenswood Ave., Menlo Park, CA. 94025

ABSTRACT

This paper addresses the problems of (1) representing natural shapes such as mountains, trees and clouds, and (2) computing such a description from image data. In order to solve these problems we must be able to relate natural surfaces to their images; this requires a good model of natural surface shapes. Fractal functions are good a choice for modeling natural surfaces because (1) many physical processes produce a fractal surface shape, (2) fractals are widely used as a graphics tool for generating natural-looking shapes, and (3) a survey of natural imagery has shown that the 3-D fractal surface model, transformed by the image formation process, furnishes an accurate description of both textured and shaded image regions. This characterization of image regions has been shown to be stable over transformations of scale and linear transforms of intensity.

Much work has been accomplished that is relevant to computing 3-D information from tbe image data, and the computation of a 3-D fractal-based representation from actual image data has been demonstrated using an image of a mountain. This example shows the potential of a fractal-based representation for efficiently computing good 3-D representations of natural shapes, including such seemingly-difficult cases as mountains, clumps of leaves and clouds.

1. INTRODUCTION

This paper addresses two related problems: (1) representing natural shapes such as mountains, trees and clouds, and (2) computing such a description from image data. The first step towards solving these problems, it appears, is to obtain a model of natural surface shapes. The task of finding such a model is extremely important to computer vision because we face problems that seem impossible to address with standard descriptive techniques. How, for instance, should we describe the shape of leaves on a tree? Or grass? Or clouds? When we attempt to describe such common, natural shapes using standard shape-primitive representations, the result is an unrealistically complicated model of something that, viewed introspectively, seems very simple.

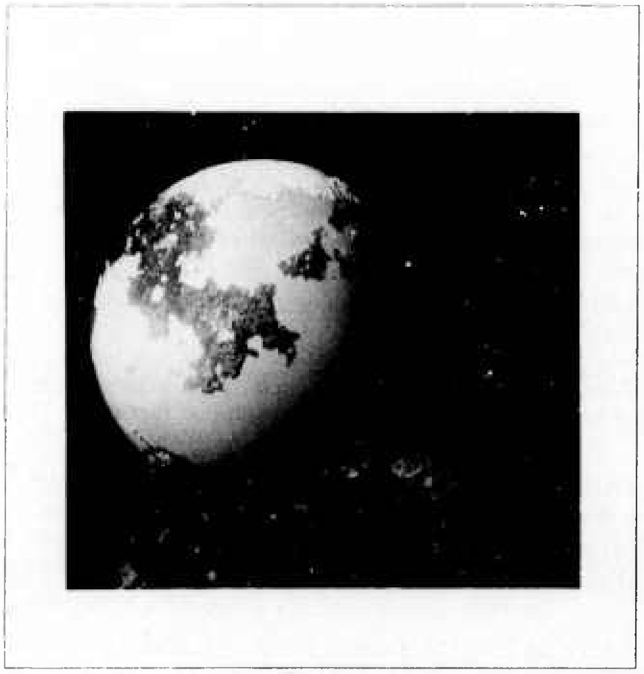


Figure 1. Fractal-based models of natural shapes, by Mandelbrot and Voss [4].

Furthermore, how can we extract 3-D information from the image of a textured surface when we have no models that describe natural surfaces and how they evidence themselves in the image? The lack of such a 3-D model has generally restricted image texture descriptions to being *ad hoc* statistical measures of the image intensity surface. A good model of natural surfaces together with the physics of image formation would provide the analytical tools necessary for relating natural surfaces to their images. The ability to relate image to surface can provide the necessary leverage for dealing appropriately with the problems of finding a good representation for natural surfaces and computing such a description from the image data.

Even shape-from-shading [22,23] and surface-interpolation methods [24] are limited by the lack of a 3-D model of natural surfaces. Currently all sucb methods employ the heuristic of "smoothness" to relate neighboring points on the surface. Such heuristics are applicable to many man-made surfaces, of course, but are demonstrably untrue of most natural surfaces. In order to apply such techniques to natural surfaces, therefore, we must find a heuristic that is true of natural surfaces. Finding such a heuristic requires recourse to a 3-D model of natural surfaces.

* The research reported herein was supported by the Defense Advanced Research Projects Agency under Contract No. MDA 903-83-C-0027; this contract is monitored by the U. S. Army Engineer Topograpbic Laboratory. Approved for public release, distribution unlimited.

The *image* $\Pi(O)$ is again a union of faces. In fact, for any $x \in \partial\Pi(O)$ there is an edge e of O such that $\Pi^{-1}(\{x\}) \cap e \cap \overline{\mathfrak{S}} \neq 0$.

DEFINITION 6: The *picture* $Pict(O,\Pi)$ associated to Π, O is the collection

$$\{\Pi(e \cap \overline{\mathfrak{S}}) | e \in E(O), e \cap \overline{\mathfrak{S}} \neq 0\},$$

where $E(O)$ denotes the set of all edges of the object O.

It is easy to see that the image $\Pi(O)$ can be generated from $Pict(O,\Pi)$.

4. Facts from Images

We shall enumerate several facts about interpretations of pictures which are now provable in the formal setup given above. They can all be found, in one form or another, in Binford [1]. In the following, $e = \Pi(\hat{e} \cap \overline{\mathfrak{S}})$ and $f = \Pi(\hat{f} \cap \overline{\mathfrak{S}})$ denote arbitrary elements of $Pict(O,\Pi)$.

FACT 1: Any element $e \in Pict(O,\Pi)$ is an image of an edge of O, lying on two non-coplanar faces.

FACT 2: If $A \subset \Pi(O)$ is an open, connected set which does not intersect any set from $Pict(O,\Pi)$, then $\Pi^{-1}(A) \cap \overline{\mathfrak{S}}$ is a part of a face (in particular, contained in a plane) and $\Pi : \Pi^{-1}(A) \cap \overline{\mathfrak{S}} \rightarrow A$ is bijective.

FACT 3: If $e, f \in Pict(C,\Pi)$ are parallel, then so are $\hat{e}$ and $\hat{f}$.

FACT 4: If P, Q, R are distinct collinear points that can be expressed as intersections of elements of $Pict(O,\Pi)$, then there are collinear points p, q, r above them in $\overline{\mathfrak{S}}$.

The following three facts give a complete analysis of points of intersection in $Pict(O,\Pi)$. They depend strongly on the linearity and general position assumptions.

FACT 5: Assume $e, f \in Pict(O,\Pi)$ form a L-vertex at a point P; i.e. P is an endpoint for both of them, and they are not collinear. Then $\hat{e}$ and $\hat{f}$ intersect at a point above P.

FACT 6: Assume $e, f \in Pict(O,\Pi)$ form a T-junction at a point P; i.e. P is an endpoint for e and an interior point for f. Then $\hat{e}$ and $\hat{f}$ do not intersect; in fact $\hat{e}$ contains a point above $\hat{f}$ and P.

FACT 7: Assume $e, f \in Pict(O,\Pi)$ form a X-vertex at a point P; i.e. P is an interior point for e, f. Then $\hat{e}$ and $\hat{f}$ intersect at a point above P.

These facts are quite sufficient to determine the impossibility of say, the Penrose triangle (see f.ex. Binford [1], figure 6). In that case the image can be proved to depict three planar regions bordered by three lines which do not have a common point of intersection.

5. References

[1] T.Binford, Inferring Surfaces from Images, AI Journal, (August 1981).

[2] M.B.Clowes, On Seeing Things, Artificial Intelligence 2, (1971), 79-116.

[3] D.A.Huffman, Impossible Objects as Nonsense Sentences, Machine Intelligence 6, (1971), 295-324.

[4] G.Markowsky and M.A.Wesley, Fleshing Out Projections, IBM Journal of Research and Development 25, (1981), 934-954.

[5] D.Mumford, Algebraic Geometry I Complex Projective Varieties, Grundlehren der mathematischen Wissenschaften 221, Springer-Verlag (1976).

[6] C.P.Rourke and B.J.Sanderson, Introduction to Piecewise-Linear Topology, Ergebnisse der Mathematik 69, Springer-Verlag (1972).

[7] D.Waltz, Understanding Line Drawings of Scenes with Shadows, in *The Psychology of Computer Vision*, McGraw-Hill Book Co., New York, (1975), 19-91.

Research supported by NSF grants MCS-82-06565, MCS-79-05998 and ARPA contract N00039-82-C-0250.

Fractal functions seem to provide such a model of natural surface shapes. Fractals are a novel class of naturally-arising functions, discovered primarily by Benoit Mandelbrot. Mandelbrot and others [1,2,4] have shown that fractals are found widely in nature and that a number of basic physical processes, such as erosion and aggregation, produce fractal surfaces. Because fractals look *natural* to human beings, much recent computer graphics research has focused on using fractal processes to simulate natural shapes and textures (see Figure 1), including mountains, clouds, water, plants, trees, and primitive animals [3,4,5,6,7]. Additionally, we have recently conducted a survey of natural imagery and found that a fractal model of imaged 3-D surfaces furnishes an accurate description of both textured and shaded image regions, thus providing validation of this physics-derived model for both image texture and shading [19].

2. FRACTALS AND THE FRACTAL MODEL

During the last twenty years, Benoit B. Mandelbrot has developed and popularized a relatively novel class of mathematical functions known as *fractals* [1,4]. Fractals are found widely in nature [1,2,4]. Mandelbrot shows that a number of basic physical processes, ranging from the aggregation of galaxies to the curdling of cheese, produce fractal surfaces. One general characterization is that any process that acts locally to produce a permanent change in shape will, after innumerable repetitions, result in a fractal surface. Examples are erosion, turbulent flow (e.g., of rivers or lava) and aggregation (e.g., galaxy formation, meteorite accretion, and snowflake growth). Fractals have also been widely and successfully used to generate realistic scenes (see Figure 1), including mountains, clouds, water, plants, trees, and primitive animals [3,4,5,6,7].

Perhaps the most familiar examples of naturally occurring fractal curves are coastlines. When we examine a coastline (as in Figure 1), we see a familiar scalloped curve formed by innumerable bays and peninsulas. If we then examine a finer-scale map of the same region, we shall again see the same type of curve. It turns out that this characteristic scalloping is present at *all* scales of examination [2], i.e., the statistics of the curve are invariant with respect to transformations of scale. This fact causes problems when we attempt to measure the length of the coastline, because it turns out that the length we are measuring depends not only on the coastline but also on the length of the measurement tool itself [2]! This is because, whatever the size measuring tool selected, all of the curve length attributable to features smaller than the size of the measuring tool will be missed. Mandelbrot pointed out that, if we generalize the notion of dimension to include *fractional* dimensions (from which we get the word "fractal"), we can obtain a consistent measurement of the coastline's length.

The definition. A fractal is defined as a set for which the Hausdorff-Besicovich dimension is strictly larger than the topological dimension. Topological dimension corresponds to the standard, intuitive definition of "dimension." Hausdorff-Besicovich dimension D, also referred to as the *fractal dimension*, may be illustrated (and roughly defined) by the examples (1) of measuring the length of an island's coastline, and (2) measuring the area of the island.

To measure the length of the coastline we might select a measuring stick of length λ and determine that n such measuring sticks could be placed end to end along the coastline. The length of the coastline is then intuitively $n\lambda$. If we were measuring the area of the island, we could use a square of area λ^2 to derive an area of $m\lambda^2$, where m is the number of squares it takes to cover the island. If we actually did this, we would find that both of these measurements vary with λ, the length of the measuring instrument — an undesirable result.

In these two examples the length λ is raised to a particular power: the power of one to measure length, the power of two to measure area. These are two examples of the general rule of raising λ to a power that is the *dimension* of the object being measured. In the case of the island, raising λ to the topological dimension does not yield consistent results. If, however, we were to use the power 1.2 instead of 1.0 to measure the length, and 2.1 instead of 2.0 to measure the area, we would find that the measured length and area remained constant regardless of the size of the measuring instrument chosen.* The positive real number D that yields such a consistent measurement is the *fractal dimension*. D is always greater than or equal to the topological dimension.

The most important lesson the work of Mandelbrot and others teaches us is the following:

Standard notions of length and area do not produce consistent measurements for many natural shapes: the basic metric properties of these shapes vary as a function of the fractal dimension. Fractal dimension, therefore, is a *necessary* part of any consistent description of such shapes.

This result, which could almost be stated as a theorem, demonstrates the *fundamental* importance of knowing the fractal dimension of a surface. It implies that *any* description of a natural shape that does not include the fractal dimension cannot be relied upon to be correct at more than one scale of examination.

Fractal Brownian functions. Virtually all the fractals encountered in physical models have two additional properties: (1) each segment is statistically similar to all others; (2) they are statistically invariant over wide transformations of scale. Motion of a particle undergoing Brownian motion is the canonical example of this type of fractal. The discussion that follows will be devoted exclusively to fractal Brownian functions, a generalization of Brownian motion.

A random function $B(x)$ is a fractal Brownian function if for all x and Δx

$$Pr\left(\frac{B(x+\Delta x)-B(x)}{|\Delta x|^{H}} < y\right) = F(y) \qquad (1)$$

where $F(y)$ is a cumulative distribution function [1]. The fractal

*This example is discussed at greater length in Mandelbrot's book, "Fractals: Form, Chance and Dimension." The empirical data are from Richardson 1961.

dimension D of the graph described by $B(x)$ is

$$D = 2 - H \tag{2}$$

If $H = 1/2$ and $F(y)$ is a zero-mean Gaussian with unit variance then $B(x)$ is the classical Brownian function. This definition has obvious extensions to two or more topological dimensions. The fractal dimension of a fractal Brownian fuuction can also be measured from its Fourier power spectrum, as the spectral density of a fractal Brownian function is proportional to f^{-2H-1}. Discussion of the rather technical proof of this fact may be found in [1].

The fractal dimension of a surface corresponds roughly to our intuitive notion of jaggedness. Thus, if we were to generate a series of scenes with the same 3-D relief but increasing fractal dimension D, we would obtain the following sequence: first, a flat plane ($D \approx 2$), then rolling countryside ($D \approx 2.1$), a worn, old mountain range ($D \approx 2.3$), a young, rugged mountain range ($D \approx 2.5$), and finally a stalagmite-covered plane ($D \approx 2.8$).

The fractal dimension of a surface is invariant with respect to transformations of scale, as Δx is independent of H and $F(y)$. The fractal dimension is also invariant with respect to linear transformations of the data and thus it remains stable over smooth, monotonic transformations.

2.1 Fractals And The Imaging Process

Before we can use a fractal model of natural surfaces to help us understand images, however, we must determine how the imaging process maps a fractal surface shape into an image intensity surface. The mathematics of this problem is difficult and no complete solution has as yet been achieved. Nonetheless, simulation of the imaging process with a variety of fractal surface models can provide us with an empirical answer — i.e., that images of fractal surfaces are themselves fractal as long as the fractal-generating function is spatially isotropic [19]. It is worth noting that practical fractal-generation techniques, such as those used in computer graphics, have had to constrain the fractal generating function to be isotropic so that realistic imagery could be obtained [3].

Real images do not, of course, appear fractal over all possible scales of examination. The overall size of the imaged surface places an upper limit on the range of scales for which the surface shape appears to be fractal, and a lower limit is set by the size of the surface's constituent particles. In between these limits, however, we may use Equation (1) to obtain a useful description of the surface.

Simulation shows that the fractal dimension of the physical surface dictates the fractal dimension of the image intensity surface; it appears that the fractal dimension of the image is a logarithmic function of the fractal dimension of the surface. If we assume that the surface is homogeneous, therefore, we can estimate the fractal dimension of the surface by measuring the fractal dimension of the image data. Even if the surface is not homogeneous, we can still infer the fractal dimension of the surface from imaged surface contours and bounding contours, by use of Mandelbrot's results.

What we have developed, then, is a method for inferring a basic property of the 3-D surface (its fractal dimension) from the image data. The fact that the fractal dimension corresponds closely to our intuitive notion of roughness shows the importance of the measurement: we can now discover from the image data whether the 3-D surface is rough or smooth, isotropic or anisotropic. We can know, in effect, what kind of cloth the surface was cut from. The fact that the fractal dimension also describes the basic metric properties of the imaged surface is further indication that it is a critical element in any consistent representation of natural surfaces.

2.2 Applicability Of The Fractal Model

An implication of the fractal surface model is that the image intensity surface is itself fractal — and *vice versa*. This is because image intensity is primarily a function of the angle between the surface normal and the incident illumination; thus, if the image intensities satisfy Equation (1), then (for a homogeneous surface) the angle between surface normal and illuminant must also and, integrating, we find that the 3-D surface is a spatially isotropic fractal.

A method of evaluating the usefulness of the fractal surface model, therefore, is to determine whether or not images of natural surfaces are well described by a fractal function. To evaluate the applicability of the fractal model, we first rewrite Equation (1) to obtain the following description of the manner in which the second-order statistics of the image change with scale:

$$E(|dI_{\Delta x}|)\|\Delta x\|^{-H} = E(|dI_1|) \tag{3}$$

where k is a constant and $E(dI_{\Delta x})$ is the expected value of the change in intensity over distance Δx. Equation (3) is a hypothesized relation among the image intensities; a hypothesis that we may test statistically. If we find that Equation (3) is true of the image intensity surface and the viewed surface is homogeneous and continuous then we may conclude that the 3-D surface is itself fractal. It is an important characteristic of the fractal model that we can determine its appropriateness for particular image data because it means that we can know when, and when *not*, to use the model.

To evaluate the suitability of a fractal model for natural textures, the homogeneous regions from each of six images of natural scenes were densely sampled. In addition, twelve textures taken from Brodatz [8] were digitized and examined (see Figure 3). The intensity values within each of these regions were then approximated by a fractal Brownian function and the approximation error observed.

For the majority of the textures examined (77%), the model described the image data accurately (see [19] for more detail). In 15% of the cases the region was constant except for random, zero-mean perturbations; consequently, the fractal function correctly approximates the image data, although the fractal dimension was equal to the topological dimension and thus the data's dimension is technically not "fractional." The fit was poor in only 8% of the regions examined and, in many of these cases, it appeared that the image digitization had become saturated.

The fact that the vast majority of the regions examined were quite well approximated by a fractal Brownian function indicates that the fractal surface model will provide a useful description of natural surfaces and their images. Fractal Brownian functions do *not*, of course, account for such large-scale spatial structure as those seen in the image of a brick wall or a tiled floor. Such structures must be accounted for by other means.

3. INFERRING SURFACE PROPERTIES

Fractal functions appear to provide a good description of natural surface textures and their images; thus, it is natural to use the fractal model for texture segmentation, classification and shape-from-texture. The first four headings of this section describe the research that has been performed in this area, and indicate likely directions for further research.

Fractal functions with $H \approx 0$ can be used to model smooth surfaces and their reflectance properties. For the first time, therefore, we can offer a single model encompassing both image shading and texture, with shading as a limiting case in the spectrum of texture granularity. The fractal model thus allows us to make a reasonable and rigorous definition of the categories "texture" and "shading," thus enabling us to discover similarities and differences between them. The final heading of this section briefly discusses this result.

3.1 An Example Of Texture Segmentation

Figure 2(a) shows an aerial view of San Francisco Bay. This image was digitized and the fractal dimension computed for each 8×8 block of pixels. Figure 2(b) shows a histogram of the fractal dimensions computed over the whole image. This histogram of fractal dimension was then broken at the "valleys" between the modes of the histogram, and the image segmented into pixel neighborhoods belonging to one mode or another.[*] Figure 2(c) shows the segmentation obtained by thresholding at the breakpoint indicated by the arrow under (b); each pixel in (c) corresponds to an 8×8 block of pixels in the original image. As can be seen, a good segmentation into water and land was achieved — one that cannot be obtained by thresholding on image intensity.

This image was then averaged down, from 512×512 pixels into 256×256 and 128×128 pixel images, and the fractal dimension recomputed for each of the reduced images. Figures 4 (d) and (e) illustrate the segmeutations produced by using the *same* breakpoint as had been employed in the original full-resolution segmentation. These results demonstrate the stability of the fractal dimension measure across wide (4 : 1) variations in scale.

Several other images have been segmented in this manner [19]. In each case a good segmentation was achieved. The computed fractal dimension, and thus the segmentation, was found to be stable over at least 4 : 1 variations in scale; most were stable over a range of 8 : 1. Stability of the fractal description is to be expected, because the fractal dimension of the image is directly related to the fractal dimension of the viewed surface, which is a property of natural surfaces that has been shown to be invariant with respect to transformations of scale [2].

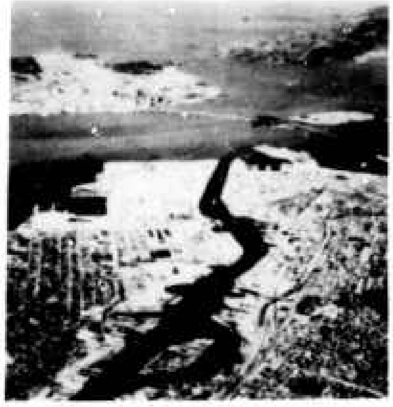


Figure 2. San Francisco Bay, and its texture segmentations.

The fact that the fractal description of texture is stable with respect to scale is a critically important property. After all, consider: **how can we hope to compute a stable, viewer-independent representation of the world if our information about the world is not stable with respect to scale?** This example of texture property measurement reiterates what we observed earlier, i.e., the fact that the fractal dimension of the surface is *necessary* to any consistent description of a natural surface.

3.2 A Comparison With Other Segmentation Techniques

To obtain an objective comparison with previously established texture segmentation techniques, a mosaic of eight natural textures taken from Brodatz [8] was redigitized. The digitized texture mosaic, shown in Figure 3, was constructed by Laws [9,10] for the purpose of comparing various texture segmentation procedures. The textures that comprise this data set were chosen to be as visually similar as possible; gross statistical differences were removed by mean-value- and histogram-equalization.

Segmentation performance for these data exists for several techniques and, although differences in digitization complicate any comparisons we might wish to make, Laws's performance figures nevertheless serve as a useful yardstick for assessing performance on this data.

For this comparison simple orientational information was incorporated into the fractal descriptiou; the fractal dimension was calculated separately for the x and y coordinates. The two-parameter fractal segmenter yielded a theoretical classifieation accuracy of 84.4%. This compares quite favorably with correlation techniques [11,12] reported by Laws as attaining 65% ac-

[*] No attempt was made to incorporate orientational information into measurement of the local fractal dimension, i.e., differences in dimension among various image directions at a point were collapsed into one average measurement.

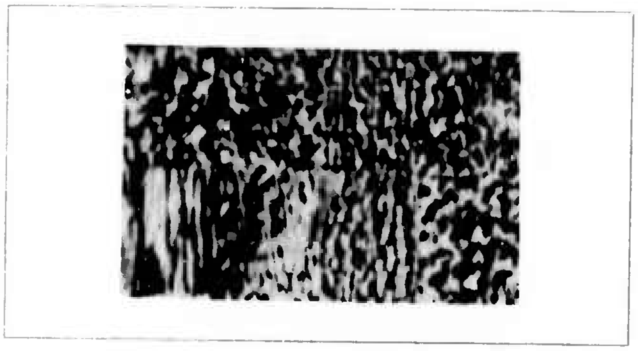


Figure 3. The Brodatz textures used for comparison.

curacy, as well as with co-occurrence techniques [13,14] reported to be 72% accurate. This superior performance was achieved despite the large number of texture features employed by the other methods.

The simple two-parameter fractal segmenter even compares well with Laws's own texture energy statistics; even though his segmentation procedure included more than a dozen texture statistics that were optimized for the test data, its theoretical segmentation accuracy was only 3% better. Thus, the results of this comparison indicate that fractal-based texture segmentation will likely prove to be a general and powerful technique (for more details, see [19]).

3.3 Relationship To Texture Models

The fact that the fractal dimension of the image data can be measured by using either co-occurrence statistics in conjunction with Equation (1), or by means of the Fourier power spectrum, suggests one interesting aspect of the fractal model: it highlights a formal link between co-occurrence texture measures [13,14] and Fourier techniques [15,16,17]. The mathematical results Mandelbrot derives for fractal Brownian functions show that the way interpixel differences change with distance determines the rate at which the Fourier power spectrum falls off as frequency is increased, and vice versa.

Thus, it appears that the fractal model offers potential for unifying and simplifying the co-occurrence and Fourier texture descriptions. If we believe that natural surface textures and their images are fractal (as seems to be indicated by the previous results), then the fractal dimension is the most relevant parameter in differentiating among textures. In this case we would expect both the Fourier and co-occurrence techniques to provide reasonable texture segmentations, as both yield sufficient information to determine the fractal dimension. The advantage of the fractal model would be that it captures a simple physical relationship underlying the texture structure — a relationship lost with either of the other two characterizations of texture. Knowledge of the fundamental physical principle can result in both increased computational efficiency and further insight.

3.4 Shape From Texture

There are two ways surface shape is reflected in image texture: (1) projection foreshortening, a function of the angle between the viewer and the surface normal, and (2) the perspective texture gradient that is due to increasing distance between the viewer and the surface. These two phenomena are independent in that they have separate causes. Thus, they can serve to confirm each other — i.e., if projection foreshortening is used to estimate surface tilt, that estimate is ***independently confirmed*** if there is a texture gradient of the proper magnitude and same direction [17,18]. We may be confident our estimate is correct when such independent confirmation is found.

The fractal dimension found in the image appears to be nearly independent of the orientation of the surface (by virtue of independence with respect to scale); therefore fractal dimension cannot be used to measure surface orientation. Projection foreshortening does, however, affect the variance of the distribution $F(y)$ associated with the fractal dimension (see Equation (1)). Foreshortening affects $Var(F(y))$ in exactly the manner it affects the distribution of tangent direction.

Thus, to estimate surface orientation, we might assume that the surface texture is isotropic and estimate surface orientation on the basis of previously derived results [18]. While this often works [19], the necessity of assuming isotropy is a serious shortcoming of this technique. An important new result, therefore, is that we may in part cure this problem by observing the fractal dimensions in the x and y directions. If they are unequal we have *prima facie* evidence of anisotropy in the surface texture, because fractal dimension is unaffected by projection.

However a foreshortening-derived estimate of surface orientation is produced, we may still seek ***confirmation*** of it by measuring the perspective texture gradient; if confirmation is found, we may be confident of our estimate. Such a gradient appears in Figure 2: the houses dwindle in size with increasing distance from the viewer. Initial results, detailed in [19], indicate that perspective texture gradients can be inferred from the locally computed fractal dimension.

This two new results, i.e., the ability to obtain evidence of surface texture anisotropy and the measurement of the perspective texture gradient, are extremely important because they offer a way to make shape-from-unfamiliar-texture techniques sufficiently reliable so as to be useful. Development of these techniques, therefore, constitute an important task for future research.

3.5 Shading And Texture

Fractal functions with $H \approx 0$ can be used to model smooth surfaces and their reflectance properties accurately. When $H \approx 0$, the surface is locally planar, except for small, random variations described by the function $F(y)$ in Equation (1). If we assume that incident light is reflected at the angle of incidence and we make the variance of $F(y)$ small relative to the pixel size, the surface will be mirrorlike. If, on the other hand, the variance of $F(y)$ is large relative to the pixel size, the surface will become more Lambertian.

The fractal model, therefore, is a single model that can account for both image shading and texture, with shading corresponding to the limiting value of H. The fractal model thus allows us to make a reasonable and rigorous definition of the categories "texture" and "shading," in terms that can be measured by using the image data. One important goal of future research will be to discover similarities or differences between these two categories; initial results indicate that local shape-from-shading results [26] can be generalized to include shape-from-texture.

4. COMPUTING A DESCRIPTION

Current methods for representing the three-dimensional world suffer from a certain awkwardness and inflexibility that makes them difficult to envisage as the basis for human-performance-level capabilities. They have encountered problems in dealing with partial knowledge or uncertain information, and they become implausibly complex when confronted with the problem of representing a crumpled newspaper, a clump of leaves or a puffy cloud. Furthermore, they seem ill-suited to solving the problem of representing a *class* of objects, or determining that a particular object is a member of that class.

What is wrong with conventional shape representations? One major problem is that they make too much information explicit. Experiments in human perception [21] lead one to believe that our representation of a crumpled newspaper (for instance) is not accurate enough to recover every z value; rather, it seems that we remember the general "crumpledness" and a few of the major features, such as the general outline. The rest of the newspaper's detailed structure is ignored; it is unimportant, *random*.

From the point of view of constructing a representation, the only important constraints on shape are the crumpledness and general outline. What we would like to do is somehow capture the notion of **constrained chance**, that is, the intuition that "a crumpled newspaper has x, y and z structural regularities and the rest is just variable detail," thus allowing us to avoid dealing with inconsequential (random) variations and to reason instead only about the structural regularities.

4.1 The Process Of Computing A Description

How shall we go about computing such a "constrained chance" description?* Let us consider the problem formally and see where that leads us. The process of computing a shape description (given some sensory data) seems best characterized as attempting to confirm or deny such hypotheses as "shape x is consistent with these sense data." Computation of a shape description, therefore, seems to be a problem in induction [20].

If, naively, we try to use an inductive method, we start with the set of all possible shape hypotheses; we then attempt to winnow the set down to a small number of hypotheses that are confirmed by the sensory data. The "set of all shape hypotheses," however, is much too large to work with. Consequently, we must take a slightly different tack.

Using the notion of constrained chance. Rather than attempting to enumerate "all shape hypotheses" explicitly, let us instead construct a *shape generator* that uses a random number generator to produce a surface shape description (I shall shortly describe how to do this). If we were to run this shape generater for an infinite period, it would eventually produce instances of every shape within a large class of shapes. If the generator were so constructed that the class of shapes produced was exactly the set of "all hypotheses" about shape, then the program for the shape generator, together with a the program for the random number generator, would comprise a description of the set of all shape hypotheses.

The shape generator illustrates how the notion of constrained chance may be used to obtain a compact description of an infinite set of shapes. By changing the constraints that determine how the output of the random number generator is translated into shape, we can change the set of shapes described; specifically, we can introduce constraints that rule out some classes of shape and thus restrict the set of shapes that are described. The ability to progressively restrict the set of shapes described allows us to use the constrained-chance shape generator as the basis for induction, rather than being forced to use the explicitly enumerated set of all shape hypotheses.

The process of computing a "constrained chance description" is straightforward. We use image data to infer (using knowledge of the physics of image formation) constraints on the shape, and then introduce those constraints into the shape generator. The end result will be a programlike description that is capable of producing all the shapes that are consistent with the image data; i.e., we shall have a description of the shapes confirmed by the image data. This, then, is the type of description we wanted: a description of shape that contains the important structural regularities that can be inferred from the image (e.g., crumpledness, outline), but one that leaves everything else as variable, random.

Some people are already doing this. Something very much like this constrained-chance representation is already being widely utilized in the computer graphics community. Natural-looking shapes are produced by a simple fractal program that recursively subdivides the region to be filled, introducing random jaggedness of appropriate magnitude at each step [3,5]. The jaggedness is determined by specifying the fractal dimension. The shapes that can be produced in this manner range from planar surfaces to mountainlike shapes, depending on the fractal dimension. Current graphics technology often employs fractal shape generators in a more constrained mode; often the overall, general shape or the boundary conditions are specified beforehand. Thus, a scene is often constructed by first specifying initial constraints on the general shape, and then using a fractal shape generator to fill in the surface with appropriately jagged (or smooth) details. The description employed in such graphics systems, therefore, is exactly a constrained-chance description: important details are specified, and everything else is left unspecified except in a qualitative manner.

This type of description bears a close relationship to surface interpolation methods (e.g., [24]). Typically, such schemes fit a smooth surface that satisfies whatever boundary conditions are available. The initial boundary conditions, together with the interpolation function, constitute a precise description of

*The term "representation" will be used to refer to the scheme for representing shapes, while the term "description" will be reserved for specific instances. Thus one can compute a description of some object; it will be a member of the class of shapes that can be accounted for within the representation.

the surface shape. Such schemes are limited to smooth surfaces, however, and therefore are incapable of dealing with most natural shapes. In contrast, a fractal-based representation allows either rough or smooth surfaces to be fit to the initial boundary conditions, depending upon tbe fractal dimensiou. This method of description, therefore, is quite capable of describing most natural surfaces — and that is why the graphics community is turning to the use of fractal-based descriptions for natural surfaces.

In order to make use of tbis type of descriptiou it is necessary to be able to specify the surface sbape iu a *qualitative* manner, i.e., how rugged is the topography? This specification of qualitative shape can be accomplished by fixing the fractal dimension. The fact that we bave recently developed a method of inferring the fractal dimension of the 3-D surface directly from the image data means that we are now able, for the first time, to actually compute a fractal or constrained-chance description of a real scene from its image.

Not only terrestrial topography has been modeled by use of a constrained-chance representatiou, but also clouds, ponds, riverbeds, snowflakes, ocean surf and stars, just to name a few examples [1,3,4,5,6,7]. Researchers have also used constrained-chance generators to produce plant shapes [1,4,6]. A very natural-looking tree can be produced by recursively applying a random number generator and simple constraints on branching geometry. In each case a random nmuber generator plus a surprisingly small number of constraints can be used to produce very good models of apparently complex natural phenomena. Thus, there is hope for extending this approach well beyond the domain of land topography.

4.2 An Example Of Computing A Description

Figure 4 illustrates an actual example of computing such a description. Figure 4(a) is an image of a real mountain. Let us suppose that we wished to use the image data to construct a three-dimensional model of the rightmost peak (arrow), perhaps for the purpose of predicting whether or not we could climb it. I will take the standard fractal technology used in the computer graphics community as the unconstrained "primal" shape generator, as it provides an apparently accurate model of a wide range of natural surfaces.

All that is necessary to construct a description of this mountain peak is to extract shape constraints from the image and insert them into the primal shape generator. The fractal dimension of the 3-D surface is the principal parameter (constraint) required by our fractal shape generator; roughly speaking, it determines the ruggedness of the surface. The fractal dimension of the 3-D surface in the region near the rightmost peak was inferred from the fractal dimension of the image intensity surface in that area [19]. Constraint on the general outline of this peak was derived from distinguished points (those with high curvature) along the boundary between sky and mountain. These two constraints, together with the shape generator, *are* a 3-D representation of this peak; the question is: how good a representation? A view of a 3-D model derived from this representation is shown in Figure 4(b). It appears that these simple constraints are sufficient for computing a good[*] 3-D representation of the peak.

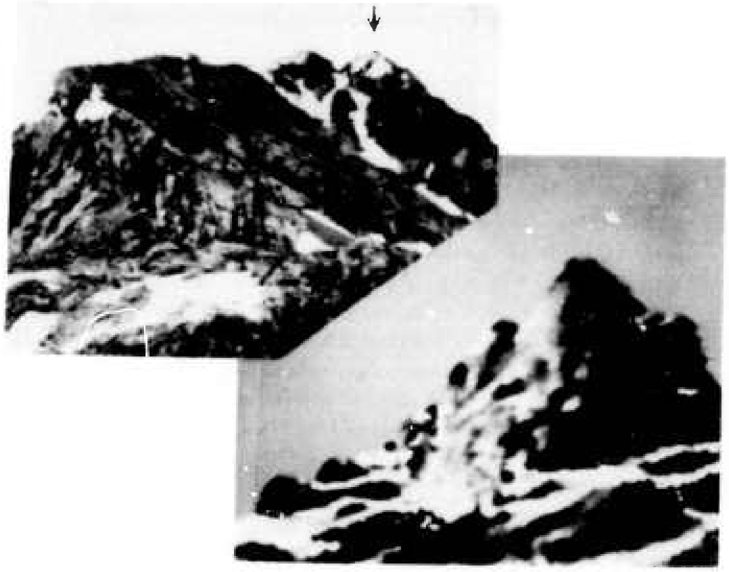


Figure 4. An example of computing a constrained-chance description.

4.3 What Do We Accomplish With This Approach?

Let's consider the problems cited above:

(1) The problem of representing a complex shape, such as a crumpled newspaper. The problem with a shape-primitive representation such as surface normals, voxcels or generalized cylinders is that the resulting description seems hopelessly complex. Because the constrained-chance representation allows us to deal only with the structural regularities and to ignore inconsequential details, the problem can become much simpler. Thus, for instance, the graphics community has found that constrained-chance fractal descriptions of complex objects (e.g., a mountain) are quite compact and easy to manipulate. It also turns out that many previously simple things, such as describing a smooth plane, remain simple.

How does this representation function when we want to compute a description of a *specific* mountain, bush or other entity from its image? Current "shape-from-x" research furnishes constraints on shape in a variety of forms: surface orientation (from texture [15 — 18,25], shading [22,23,26]), relative depth (from motion [27,28], contour [29 — 31]), and absolute depth (from stereo [32 — 34], egomotion [35,36]). It appears to be fairly straightforward to mix each of the various flavors of constraint into the vanilla-flavor shape generator [3,5], although significant research remains to be done. As more shape constraints are obtained from the image, the description becomes more and more precise; i.e., there is less and less chance in the description.

[*] Rather primitive ray tracing, etc., was used to generate this image; better code is being implemented.

Eventually, only one shape satisfies all of the constraints.

How complex could such a description become? The constrained-chance representation would *at worst* be as complex as a two-dimensional array of z values representing the same surface, because we could always use it to actually generate such an array of z values. As mentioned previously, experiments in human perception indicate that our representations are usually not accurate enough to recover every z value. The representation of a particular object, therefore, is likely to be quite a bit simpler than a full depth map.

(2) The problem of representing *classes* of shapes, such as are referred to by the terms "a mountain," or "a bush." Again, the ability to specify important structural details and leave the rest only qualitatively constrained allows simplification of the problem. The definition of "a mountain," for instance, might reasonably consist entirely of a specification of the fractal dimension of the surface and a caveat concerning size. If we are to judge by the results reported in the computer graphics literature, the notion of representation by constrained chance thus allows us, using only a few lines of code, to produce an accurate descriptiou of the *class* of shapes we label "mountains," or "bush."

(3) The problem of determining the set of appropriate descriptions when the shape is underconstrained by the sense data. The problem with standard shape-primitive representations is that either we must generate all combinations of shape primitives consistent with the sense data (a very hard problem), or pick a prototype and specify error bounds. The problem with using prototypes plus error bounds is that we are forced to overcommit ourselves by choosing the prototype; e.g., there is something seriously wrong about describing a cube as "a sphere $\pm 0.4r$", even though the cube certainly fits within the specified volume.

Because the constrained-chance representation allows details to be left constrained but unspecified, it allows us to deal with insufficient sense data by simply adding in those constraints that can be deduced from the image data and committing ourselves no further. The result is a programlike description that can be analyzed and manipulated, does not overcommit itself as to object shape, and allows examples of shapes consistent with the image data to be generated and examined.

(4) The problem of determining that a specific description is a member of a more general class. Here the problem with shape-primitive representations is that there is so much variability among the descriptions of the members of a class such as "mountain" that a description of the class as a whole seems extremely difficult, and determination of class membership even more so.

The problem of establishing class membership by using constrained-chance representations reduces to determining whether the constraints used to specify a particular description are a subset of those of the more general class. A determination regarding class membership is, therefore, exactly equivalent to determining whether one program's output is a subset of another program's output. While such automatic proof is a difficult problem, it is at least tractable and well-defined — unlike the equivalent problem can be when using a shape-primitive representation. Thus, a constrained-chance representation allows a clear and potentially useful definition of what it means to "recognize that x is an y."

Further, because we need only deal with the structural regularities, this problem can become much simpler than it might at first appear. Taking the class "a mountain" to be defined by fractal dimension and overall size (a definition that is actually sufficient to produce realistic mountain shapes) we can, for instance, easily determine that the description computed by us for the mountain peak is in fact a description of part of a mountain — a task that previously seemed to be nearly impossible.

5. SUMMARY

Fractal functions seem to provide a good model of natural surface shapes. Many basic physical processes produce fractal surfaces. Fractal surfaces also *look* like natural surfaces, and so have come into widespread uses in the computer graphics community. Furthermore, we have conducted a survey of natural imagery and found that a fractal model of imaged 3-D surfaces furnishes an accurate description of both textured and shaded image regions.

Fractal functions, therefore, are useful for addressing the related problems of representing complex natural shapes such as mountains, and computing a description of such shapes from image data. The following describes the progress achieved toward the solution of these problems.

Computing a description. Characterization of image texture by means of a fractal surface model has shed considerable light on the physical basis for several of the texture techniques currently in use, and made it possible to describe image texture in a manner that is stable over transformations of scale and linear transforms of intensity. These properties of the fractal surface model allow it to serve as the basis for an accurate image segmentation procedure that is stable over a wide range of scales.

Because fractal dimension is not affected by projection distortion, its measurement can significantly enhance our ability to estimate shape from (unfamiliar) texture. Specifically, it seems that measurement of fractal dimension can provide (1) evidence of surface texture anisotropy, and (2) an estimate of the perspective texture gradient. Both capabilities are extremely important because they provide a way to obtain independent confirmation of the assumptions on which previously-reported [18] shape-from-unfamiliar-texture techniques are based.

Representing natural shapes. A constrained-chance representation modeled after the fractal techniques used by the graphics community seems useful for representing complex natural shapes, such as a crumpled newspaper or a mountain. The problem encountered when using conventional shape-primitive representations to describe natural surfaces is that the resulting description is often hopelessly complex. Because the constrained-chance representation allows us to deal only with the structural regularities and to ignore inconsequential details, the problem can become much simpler. Thus, for instance, the graphics community has found that constrained-chance fractal descriptions of complex objects (e.g., a mountain) are quite compact and easy to manipulate. Similarly, the problem of representing *classes* of shapes, such as are referred to by the terms "a mountain," or "a bush," can also be significantly simplified.

The encouraging progress that has already been achieved on both of these problems augers well for this approach. It appears that a constrained-chance representation incorporating a fractal model of surface shape will prov.'.le an elegant solution for some of the most difficult problems encountered when attempting to progress from the image of a natural scene to its description.

REFERENCES

[1] B. B. Mandelbrot, "Fractals: Form, Chance and Dimension," W. H. Freeman and Co.,San Francisco, California, 1977.

[2] L. F. Richardson, "The Problem of Contiguity: an Appendix of Statistics of Deadly Quarrels," General Systems Yearbook, vol. 6, pp. 139-187, 1961.

[3] A. Fournier, D. Fussel and L. Carpenter, "Computer Rendering of Stochastic Models," Communications of the ACM, vol. 25, 6, pp. 371-384, 1982.

[4] B. B. Mandelbrot, "The Fractal Geometry of Nature," Freeman, San Francisco, 1982.

[5] A. Norton, "Generation and Display of Geometric Fractals in 3-D," Computer Graphics, vol. 16, 3, pp. 61-67, 1982.

[6] Y. Kawaguchi, "A Morphological Study Of The Form Of Nature," Computer Graphics, vol. 16, 3, pp. 223-232, 1982.

[7] L. C. Carpenter, "Vol Libre," Computer Generated Movie, 1980.

[8] P. Brodatz, "Textures: A Photographic Album for Artists and Designers," Dover, New York, New York, 1966.

[9] K. Laws, "Textured Image Segmentation," Report 940, USC Image Processing Institute, Los Angeles, California, 1980.

[10] D. H. Ballard and C. M. Brown, "Computer Vision," Prentice-Hall Inc., Englewood Cliffs, N.J., 1982.

[11] W. K. Pratt, O.D. Faugeras, and A. Gagalowicz, "Visual Discrimination of Stochastic Texture," IEEE Transactions on Systems, Man and Cybernetics, vol SMC-8, pp. 460-473, 1978.

[12] K. Deguchi and I. Morishita, "Texture Characterization and Texture-Based Image Partitioning Using Two-Dimensional Linear Estimation Techniques," IEEE Transactions on Computers, vol C-27, pp. 739-745, 1978.

[13] A. Rosenfeld and E. B. Troy, "Visual Texture Analysis," IEEE Conference on Feature Extraction and Analysis, pp. 115-121, Argonne, Ill. Oct 1970.

[14] R. M. Haralick, K. Shanmugam and J. Dinstein, "Textural Features for Image Classification," IEEE Transactions on Systems, Man and Cybernetics, vol. SMC-3, pp. 610-621, 1973.

[15] R. Bajacsy and L. Lieberman, "Computer Description of Real Outdoor Scenes," Proceedings of 2d International Joint Conference on Pattern Recognition, pp. 174-179, Copenhagen, Aug 1974.

[16] H. Maurer, "Texture Analysis With Fourier Series," Proceedings of the 9th International Symposium on Remo'e Sensing of the Environment, pp 1411-1420, Ann Arbor, Michigan, April 1974.

[17] R. Bajacsy and L. Lieberman, "Texture Gradient as a Depth Cue," Computer Graphics and Image Processing, vol. 5, 1, pp 52-67, 1976.

[18] A. P. Witkin, "Recovering Surface Shape and Orientation from Texture," Artificial Intelligence, 17, pp. 17-47 (1981).

[19] A. Pentland, "Fractal Textures," Proceedings of IJCAI 83, Karlsruhe, Germany, August 1983.

[20] R. L. Gregory, "Eye And Brain: The Psychology of Seeing," New York, McGraw-Hill, 1972.

[21] D. A. Norman, "Memory and Attention," New York, Wiley, 1976.

[22] B. K. P. H. Horn, "Shape From Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from One View," A.I. Technical Report 79, Project MAC, M.I.T. (1970).

[23] B. K. P. H. Horn and K. Ikeuchi, "Numerical Shape from Shading and Occluding Boundaries," Artificial Intelligence, 15, Special Issue on Computer Vision, pp. 141-184 (1981).

[24] W. E. L. Grimson, "Computing Shape Using A Theory Of Human Stereo Vision," Ph.D. Thesis, Dept. of Mathematics, M.I.T. (1980).

[25] J. R. Kender, "Shape From Texture: An Aggregation Transform that Maps a Class of Textures Into Surface Orientation," Proceedings of the Sixth International Joint Conference on Artificial Intelligence, Tokyo, Japan (1979).

[26] A. P. Pentland, "Local Computation Of Shape," Proceedings of the National Conference on Artificial Intelligence, Pittsburgh, Pennsylvania (1982).

[27] S. Ullman, "The Interpretation of Visual Motion," M.I.T. Press, Cambridge, Massachusetts (1979).

[28] D. Hoffman and B. E. Finchbaugh, "The Interpretation of Biological Motion," Bio. Cybern. 42, pp. 195-204, 1982.

[29] S. W. Zucker, R. A. Hummel, and A. Rosenfeld, "An Application of Relaxation Labeling to Line and Curve Enhancement," IEEE Transactions on Computers, C-26, 4, pp. 394-403 (1977).

[30] D. Marr, "Analysis Of Occluding Contour," Proc. Royal Soc. Lond. 197, pp. 441-475, 1977.

[31] A. P. Witkin, "Computational Theory of Line Drawing Interpretation," Artificial Intelligence Center, SRI International, Menlo Park, California (October 1981).

[32] D. C. Marr and T. Poggio, "A Computational Theory of Human Stereo Vision," Proc. R. Soc. London, 204, B, pp. 301-328 (1979).

[33] H. P. Moravec, "Rover Visual Obstacle Avoidance," Proceedings of the Seventh Joint Conference on Artificial Intelligence, pp. 785-790 (1981).

[34] H. Baker and T. O. Binford, "Depth From Edge and Intensity Based Stereo," Proceedings of the Seventh Joint Conference on Artificial Intelligence, pp. 631-636 (1981).

[35] A. Bruss and B.K.P.H. Horn, "Passive Navigation," Proceeding of the Image Understanding Workshop, Stanford, California, September 1982.

[36] K. Prazdny,"Egomotion and Relative Depth Map from Optical Flow," Technical Memo, Computer Science Department, University of Essex, Colchester, England.

Rule Based Strategies for Image Interpretation

T. E. Weymouth, J. S. Griffith, A. R. Hanson and E. M. Riseman

Department of Computer and Information Science[1]
University of Massachusetts at Amherst
Amherst, Massachusetts 01003

ABSTRACT

A rule based image interpretation system is presented which is shown to be effective in interpreting complex outdoor scenes. The system utilizes world knowledge to reduce the ambiguities in image measurements obtained from simple interpretation rules. These rules involve sets of partially redundant features each of which defines an area of feature space which represents a "vote" for an object. The features include color, texture, shape, size, image location, and relative location to other objects. Convergent evidence from multiple interpretation strategies is organized by top-down control mechanisms in the context of a partial interpretation. One such strategy extends a kernel interpretation derived through the selection of object exemplars and regions which represent the most reliable image specific hypotheses of a general object class. The use of exemplar strategies and other top-down strategies results in the extension of partial interpretations from islands of reliability.

1. Introduction

The use of world knowledge, together with top down control, is beneficial and probably essential in domains where uncertain data and intermediate results containing errors cannot be avoided. The task of "understanding" images of unconstrained natural scenes is such a domain. Ambiguity and uncertainty in image interpretation tasks arise from many sources, including the inherent variation of objects in the natural world (e.g., the size, shape, color and texture of trees), the ambiguities arising from the perspective projection of the 3D world onto a 2D image plane, occlusion, changes in lighting, changes in season, image artifacts introduced by the digitization process, etc. These difficulties are compounded by the lack of precise mechanisms or theories of visual processes that would allow the accurate and reliable extraction of important image events. Nevertheless, many real scenes contain numerous examples of instances in which human observers can infer the presence and location of objects from marginal bottom-up information.

Attempts in computer vision research to contend with these issues involve the integration of the results of analysis of different aspects of the visual data (such as color, texture, shape, perspective, stereopsis, motion, etc.), and from overlapping local contextual and environmental constraints. We will present a system here that uses a large amount of stored knowledge to carry out the image interpretation task. One can only expect these results to be improved by more effective low-level processes than those presented in this system.

Early systems demonstrated that task-specific knowledge could be used to advantage in image interpretation (for example [9, 10, and 11]). Recent research has been directed towards developing increasingly general representations of the world knowledge needed for image understanding. For example, generalized cylinders, a fairly robust representation of form, are used in the system developed by Brooks [2]. In this representation, the metric relations among objects and within object descriptions are parameterized. The image interpretation process builds a graph representaiton of the particular image, and also fixes bounds on those free parameters needed for interpretation. However, the object representation and the matching process do not make use of any image features other than ribbons, a linked set of edges that are candidates for projections of generalized cylinders.

The systems developed by Ohta [7] for understanding images of buildings in outdoor settings and by Nagao [6] for understanding aerial photographs make much wider use of the image data. In both these systems the objects are described in terms of their possible image appearance; these descriptions include both spectral and spatial features. Both systems have a very rich description of appearance but little description of form. The system developed by Nagao

[1] This research was supported in part by the Defense Advanced Research Projects Agency under contract number N00014-82-K-046, the National Science Foundation under grant number MCS-7918209, and by the Air Force Office of Scientific Research.

is of special interest here, because of his use of "characteristic regions." By identifying those regions that can be given a tentative identity (as, say, "a large textured region") with a high degree of certainty and by subsequently associating a label with those regions ("forest"), the identification process can build up "islands of certainty" that yield information about the appearance of specific objects in the image. This is similar to the concept of object exemplars described in Section 4. A review of these and other related work appears in [1].

In this paper the interpretation task examined is that of labelling an initial region segmentation of an image with object (and object part) labels, when the image is known to be a member of a restricted class of scenes (e.g., suburban house scenes). An important aspect of this task is the effective use of scene/image knowledge in the interpretation process, particularly on methods and techniques for aggregating and mapping preliminary region, boundary, and/or surface data into more abstract descriptions. The results discussed in Section 5 were obtained from a version of the VISIONS system configured with a region segmentation system, a knowledge network, such as the one shown in Figure 1, a collection of interpretation "rules", and a set of interpretation "strategies".

2. A Knowledge Network and Representation Using Schemata

Description of scenes, at various levels of detail, are captured in a set of schema hierarchies [4]. A schema graph is a data structure defining an expected collection of objects, such as a house scene, the expected visual attributes associated with the objects in the schema (each of which can have an associated schema), and the expected relations among them. For example, a house (in a house scene hierarchy) has roof and house wall as sub-parts, and the house wall has windows, shutters, and doors as sub-parts. The knowledge network of Figure 1 is a simplified version of a schema hierarchy as developed in [8]. Each schema node (e.g. house, house wall, and roof) has both a structural description appropriate to the level of detail and methods of access to a set of recognition and verification strategies called interpretation strategies. For example, the sky-object schema (associated with the outdoor-scene schema) has access to the exemplar selection and extension strategy discussed below.

In general, the information available about any scene component falls into one of three classes: knowledge of form, of spectral characteristics, and of plausible relations with other objects. Interpretation rules relate image events to knowledge events by providing evidence for or against part/sub-part hypotheses. An interpretation strategy, associated with a schema node, specifies how specific interpretation rules may be applied, and how combined results from multiple rules may be used to decide whether or not to "accept" (i.e., instantiate) an object hypothesis. The interpretation strategy thus represents both control local to the node and top-down control over the instantiation process.

Note that the goal is not to have these interpretation rules and strategies extract exactly the correct set of regions. Our philosophy is to allow incorrect, but reasonable, hypotheses to be made and to bring to bear other knowledge (such as various similarity measures and spatial constraints) to filter the incorrect hypotheses. An example of such error detection and correction in the interpretation process will be given in Section 5.

3. Rule Form for Object Hypotheses Under Uncertainty

Important schema attributes of objects include features such as color, texture, shape, relative size measurements, and expected spatial relationships with other objects, object parts, and the image frame. Unfortunately, the large variations observed in image features and the significant overlap of feature distributions across images preclude the use of standard pattern classification approaches to the problem of characterizing a set of measured features by an object label. This approach produces a large number of false-positive responses as indicated in Figure 2 for an "excess green" feature (2G-R-B).

We propose an approach to object hypothesis formation which is both simple and effective. It relies on convergent evidence from a variety of measurements and expectations. For example, in an outdoor scene taken with a camera in standard position, one would expect grass to be of medium brightness, to have a significant green component, to be located somewhere in the lower portion of the image, etc. [2] These expectations can be translated into a strategy which combines the results of many measurements into a confidence level that the region (or meta-region) represents grass.

We will illustrate the form of a simple interpretation rule based on using the expectation that grass is green. The feature used is average excess green for the region, obtained by computing the mean of 2G-R-B for all pixels in this region. Histograms of

[2] Note that a camera model and access to a 3D representation of the environment could dynamically modify the value of these location limits in the image; thus, the system would modify expectations as it orients the camera up or down relative to the ground plane.

this feature are shown in Figure 2, comparing all regions to all known grass regions across 8 samples of color outdoor scenes. An abstract version is shown in Figure 3. The basic idea is to form a mapping from a measured value of the feature obtained from an image region, say f_I, into a "vote" for the object on the basis of this single feature. One approach to defining this mapping is based on the notion of prototype vectors and the distance from a given measurement to the prototype, a well-known technique which extends to N-dimensional feature space [4]. In our case rather than using this distance to "classify", we translate it into a "vote".

Let $d(f_P,f_I)$ be the distance between the prototype feature point f_P and the measured feature value f_I. The response R of the rule is then

$$P(f_I) = \begin{cases} 1 & \text{if} \quad d(f_P,f_I) \leq \Theta_1 \\ \dfrac{\Theta_2 - d(f_P,f_I)}{\Theta_2 - \Theta_1} & \text{if } \Theta_1 < d(f_P,f_I) \leq \Theta_2 \\ 0 & \text{if } \Theta_2 < d(f_P,f_I) \leq \Theta_3 \\ -\infty & \text{if } \Theta_3 < d(f_P,f_I) \end{cases}$$

The thresholds Θ_1,Θ_2, and Θ_3 represent a gross interpretation of the distance measurements. Θ_3 allows strong negative votes if the measured feature value implies that the hypothesized object cannot be correct. For example, fairly negative values of the excess green feature imply a color which should veto the grass label. Thus, certain measurements can exclude object labels; this proves to be a very effective mechanism for filtering many spurious weak responses. Of course there is the danger of excluding the proper label due to a single feature value, even in the face of strong support from many other features. In the actual implementation of this rule form, Θ_1, Θ_2, and Θ_3 are replaced with six values so that non-symmetric rules may be defined as shown in Figure 4. There are many ways to combine the individual feature responses into a score; here we have used a simple weighted average.

4. Exemplars and Islands of Reliability

The extreme variations that occur across images can be compensated for somewhat by utilizing an adaptive strategy. This approach is based on the observation that the variation in the appearance of objects (region feature measures across images) is much greater than object variations within an image (see Figure 2).

In the initial stages, there are few if any image hypotheses, and development of a partial interpretation must rely primarily on general knowledge of expected object characteristics in the image and not on the relationship to other hypotheses. The most reliable object hypotheses can be formed using interpretation rules based on prototype matching and this can be the basis of adaptation. A largely incomplete kernel interpretation is formed based on the most reliable of these hypotheses; this forms the initial context for further interpretation strategies. One such strategy extends the kernel interpretation by using the features of labelled regions (color, texture, shape, location, and size) as "exemplars" (new prototypes) which can be used to select and label other regions of the same identity. This is similar to the method in [6], where "characteristic regions" were used to guide hypothesis formation in the early stages of interpretation. Finally a verification phase can be applied where relations between object hypotheses are examined for consistency. Thus, the interpretation is extended through matching and processing of region characteristics as well as semantic inference.

Exemplar hypothesis regions are selected by a rule of the general form described in section 3. The goal is to find a representative region that matches as closely as possible the predefined template. Once found this region (or set of regions) can be used to define an image specific template (perhaps in a different sub-space of the feature space than was used to select it). Exemplar hypotheses differ from general hypothesis rules in that they are more conservative; they should minimize the number of false hypotheses at the risk missing true target regions by narrowing their range of acceptable responses. If all regions are vetoed, secondary strategies are invoked; for example, the veto ranges can be relaxed, admitting less reliable exemplars. Figure 5 compares the results of the grass exemplar rule with the general grass hypothesis rule. The strategy can also be used to generate lists of hypotheses ordered by reliability.

The advantages of using object exemplars include:

1) an effective means for extending **reliable hypotheses** to regions which are more ambiguous; this is similar to the notion of **"islands of reliability"** [3];

2) a **knowledge-directed** technique for partially dealing with the unavoidable **region fragmentation** that occurs with any segmentation algorithm or low-level image transformation/grouping; regions that are "similar" to the exemplar can be both labelled and merged; similarity criteria can be context-sensitive so that regions will be compared to the exemplar in terms of the range of each feature of that object;

3) exemplars play a natural role in the implementation of an **hypothesize-and-verify**

control strategy; hypotheses are formed based upon initial feature information and subsequently can be used in a verification process where the relationship between labelled regions provides consistency checks on the hypotheses and the evolving interpretation.

Let us briefly consider a few of the many ways that exemplars can be used to extend object hypotheses. The similarity of region color and texture can be used to extend an object label to other regions, possibly under spatial constraints. Thus, a sky exemplar region would be restricted to comparisons with regions above the horizon which look similar to the largest, bluest region located near the top of the picture. A house wall showing through foliage can be matched to the unoccluded visible portion based upon color similarity and spatial constraints derived from inferences from house wall geometry.

The shape and/or size of a region can be used to detect other instances of multiple objects, as in the case of finding one shutter or window of a house, or one tire of a car, or one car on a road. In many situations, multiple instances of an object can be expected to have a similar size and shape. This, together with constraints on the image location, permits reliable hypotheses to be formed even with high degrees of partial occlusion. If one is viewing a house from a viewpoint approximately perpendicular to the front wall, other shutters can be found via the presence of a single shutter since there are also strong spatial constraints on their location. If two shutters are found then perspective distortion can be taken into account when looking for the other shutters, even without a camera model, under an assumption that the tops and bottoms of the set of shutters lie on a straight line [5].

5. Results of Rule Based Image Interpretation

Experiments are being conducted on a set of fifteen "house scene" images. Thus far, we have been able to extract sky, grass, and foliage (trees and bushes) from nine house images with reasonable effectiveness, and have been successful in identifying houses and their parts, including shutters (or windows), house wall and roof in three of these images. The interpretation strategies use many redundant features, each of which can very often be expected to be present. The premise is that many redundant features allow any single feature to be unreliable. The features utilized vary across color and texture attributes, shape, size, location in the image, relative location to identified objects, and similarity in color and texture to identified objects. Object hypothesis rules were employed as described in previous sections, and additional object verification rules requiring consistent relationships with other object labels are being developed. The final results shown in Figure 6 are an interpretation based on coarse segmentations. Further work on segmentation (Figure 7) is being carried out, as is the refinement of the exemplar selection and matching rules (that were shown in section 3).

An extremely important capability for an interpretation system is feedback to lower level processes for a variety of purposes. The interpretation processes should have focus-of-attention mechanisms for correction of segmentation errors, extraction of finer image detail, and verification of semantic hypotheses. An example of the effectiveness of semantically directed feedback to segmentation processes is shown in Figure 8. Two different segmentations are shown; the second, with less image detail, was used here. There is a key missing boundary between the house wall and sky which leads to incorrect object hypotheses based upon local interpretation strategies. The region is hypothesized to be sky by the sky strategy, while application of the house wall strategy (using the roof and shutters as spatial constraints on the location of house wall) leads to a wall hypothesis.

There is evidence available that some form of error has occurred in this example: 1) conflicting labels are produced for the same region by local interpretation strategies; 2) the house wall label is associated with regions above the roof (while there are houses with a wall above a lower roof, the geometric consistency of the object shape is not satisfied in this example); and 3) the sky extends down close to the approximate horizon line in only a portion of the image (which is possible but worthy of closer inspection).

In this case resegmentation of the sky-housewall region with segmentation parameters set to extract finer detail produces the results shown in Figure 8a. Subsequent remerging of similar regions produces a usable segmentation of this region as shown in 8b. It should be pointed out that in this image there is a discernable boundary between the sky and house wall. Initially, the segmentation parameters may be set so that the initial segmentation misses this boundary. This may occur because of computational requirements (fast, coarse segmentations) or as an explicit control However, once it is resegmented with an intent of overfragmentation, this boundary can be detected. Remerging based on region means and variances of a set of features allows much of the overfragmentation to be removed. Now, the same interpretation strategy used eariler produces quite acceptable results shown in Figure 9.

The current development of interpretation strategies involves the utilization of stored knowledge and a partial model (labelled regions) for hypothesis extension. In these strategies the knowledge network is examined for objects that can be inferred from identified objects, and for relations that would differentiate them. For example, the bush regions can be differentiated from other foliage based on their spatial relations to the house, and front and side house walls can be differentiated using geometric knowledge of house structure (e.g., relations between roof and walls), as shown in Figure 10. In the full system, these rules would not work in isolation as shown here, and the errors made by this type of rule would be filtered by other constraints.

Future work is directed towards refinement of the segmentation algorithms, object hypothesis rules, object verification rules, and interpretation strategies. System development is aimed towards more robust methods of control: automatic schema and strategy selection, interpretation of images under more than one general class of schema, and automatic focus of attention mechanisms and error-correcting strategies for resolving interpretation errors.

References

[1] Binford, T., "Survey of Model Based Image Analysis Systems," *International Journal of Robotics Research*, Volume 1, Number 1, Spring 1982, pp. 18-64.

[2] Brooks, R., "Symbolic Reasoning Among 3-D Models and 2-D Images," Stanford Technical Report STAN-CS-81-861 and AIM-343., Department of Computer Science, Stanford University, Stanford, California, June 1981.

[3] Erman, L., Hayes-Roth, F., Lesser, V. and Reddy, D., "The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty," *Computing Surveys*, 12(2), June 1980, pp. 213-253.

[4] Hanson, A. and Riseman, E., "VISIONS: A Computer System for Interpreting Scenes," in *Computer Vision Systems* (A. Hanson and E. Riseman, eds.), Academic Press, 1978, pp. 303-333.

[5] McCormick, C.H., "Strategies as Knowledge-Based Control in Image Interpretation," COINS Technical Report, University of Massachusetts, Amherst, MA., 1982.

[6] Nagao, M. and Matsuyama, T., *A Structural Analysis of Complex Aerial Photographs*, Plenum Press, New York, 1980.

[7] Ohta, Y., "A Region-Oriented Image-Analysis System by Computer," Ph.D. Thesis, Kyoto University, Department of Information Science, Kyoto, Japan, 1980.

[8] Parma, C., Hanson, A. and Riseman, E., "Experiments in Schema-Driven Interpretation of a Natural Scene," COINS Technical Report number 80-10, University of Massachusetts, 1980. Also in *Nato Advanced Study Institute on Digital Image Processing* (R. Haralick and J.C. Simon, eds.), Bonas, France, 1980.

[9] Rosenthal, D., "An Inquiry Driven Computer Vision System Based on Visual and Conceptual Hierarchies," Ph.D. Dissertation, The Moore School of Electrical Engineering, University of Pennsylvania, 1978.

[10] Sloan, K., "World Model Driven Recognition of Natural Scenes," Ph.D. Thesis, Moore School of Electrical Engineering, University of Pennsylvania, 1977.

[11] Tenenbaum, J. M. and Barrow, H., "Experiments in Interpretation-Guided Segmentation," Technical note 123, AI Center, Stanford Research Institute, 1976.

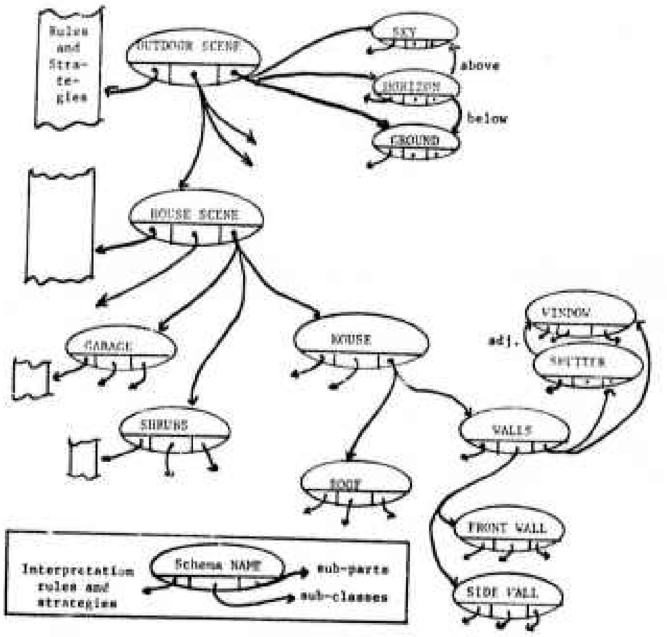
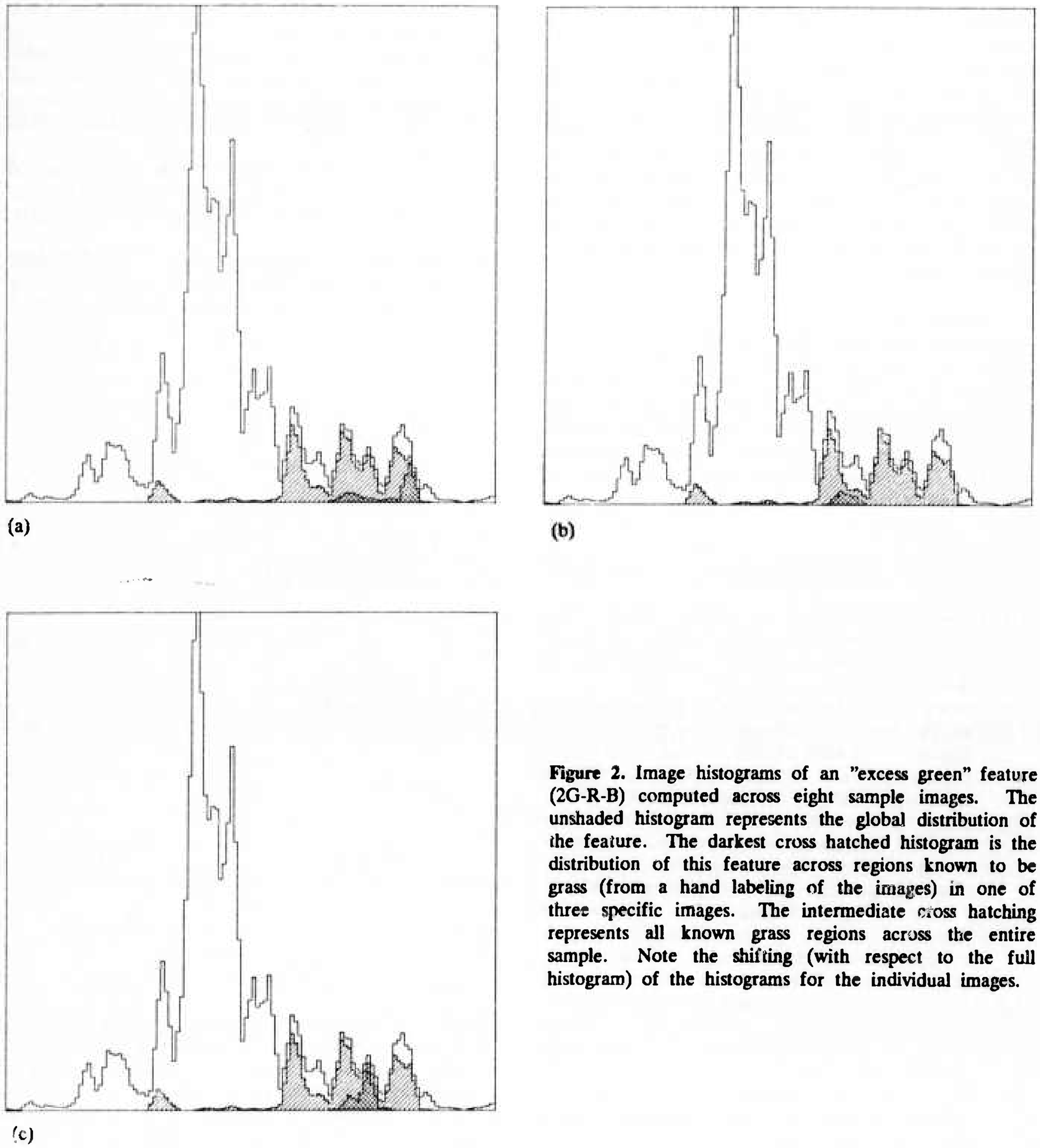


Figure 1. Abstract representation of a portion of the knowledge network used to produce the interpretation results. Knowledge about a class of scenes is represented by a hierarchy of schemata embeded in a network. Structural descriptions of each schema are organized by the component descriptions of subclass and subpart, and by spatial relations. Each schema node has acess to a set of interpretation rules which form hypotheses on the bases of image measurements, and interpretation strategies which describe how these hypotheses are combined with information in the network to form a consistent interpretation. Although every node in the network is considered to be a schema, only selected nodes in the figure show the full structure.



(a)

(b)

(c)

Figure 2. Image histograms of an "excess green" feature (2G-R-B) computed across eight sample images. The unshaded histogram represents the global distribution of the feature. The darkest cross hatched histogram is the distribution of this feature across regions known to be grass (from a hand labeling of the images) in one of three specific images. The intermediate cross hatching represents all known grass regions across the entire sample. Note the shifting (with respect to the full histogram) of the histograms for the individual images.

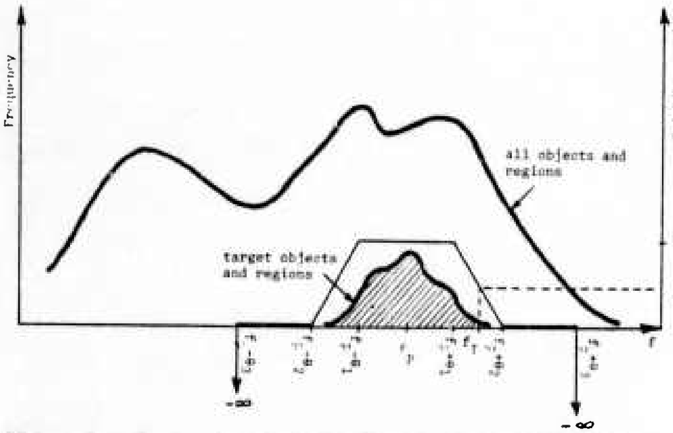


Figure 3. Structure of a simple rule for mapping an image feature measurement f_I into support for a label hypothesis on the basis of a prototype feature value obtained from the combined histograms of labeled regions across image samples. The object specific mapping is parameterized by four values, f_P, θ_1, θ_2, θ_3, and stored in the knowledge network. The use of six values will allow an asymmetric response function.

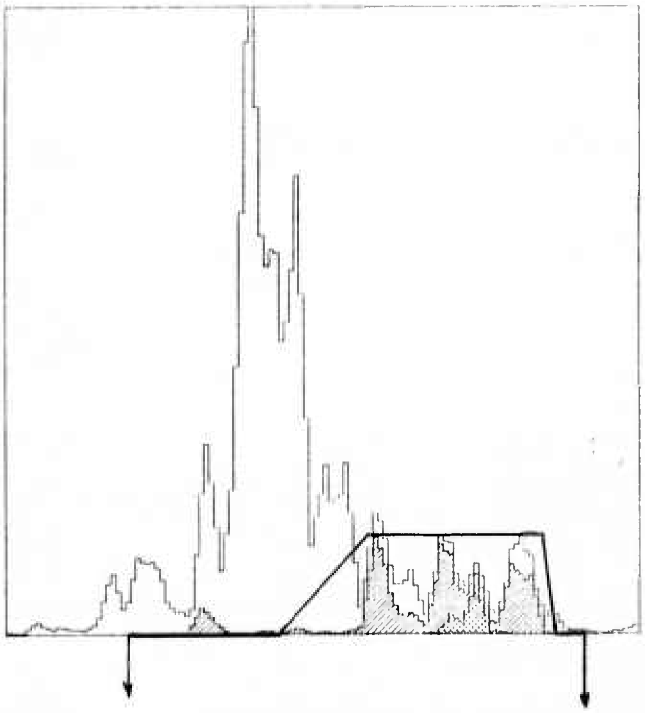
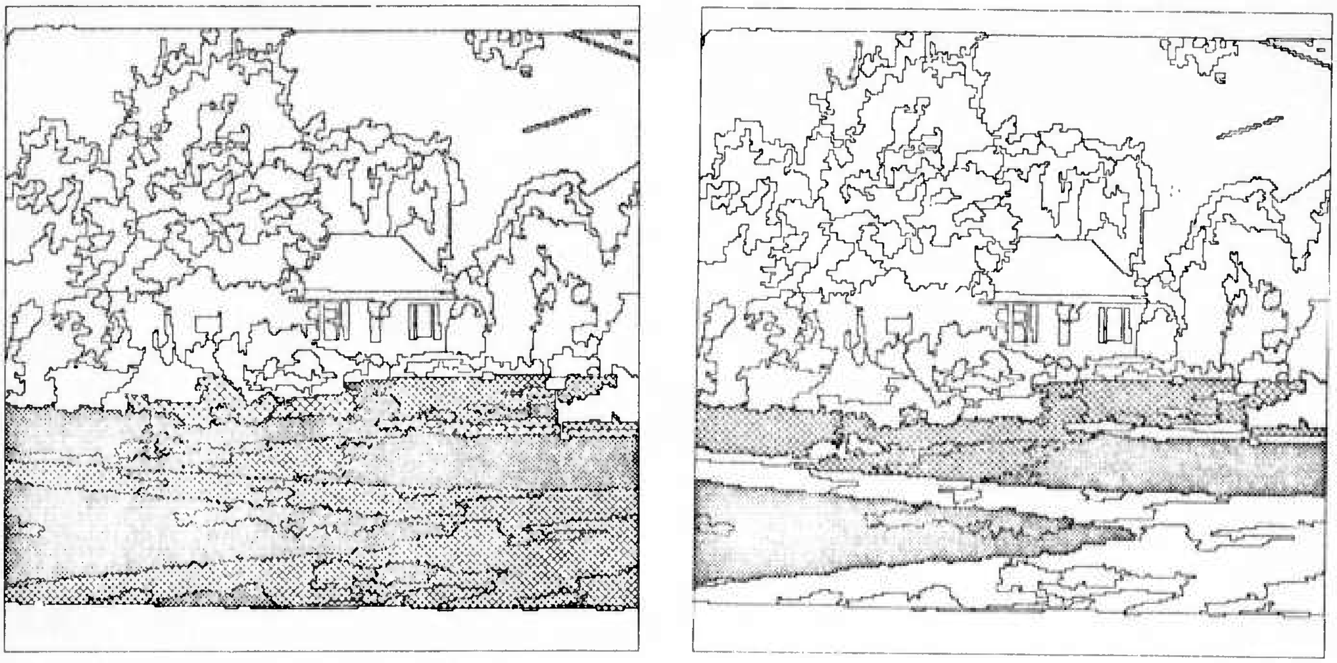


Figure 4. An example grass rule, showing an asymmetrical structure, superimposed on the histogram of Figure 2c.

Figure 5. The exemplar hypothesis rule is more selective than the corresponding general interpretation rule (based on a less selective rule form). Figure 5a shows the general grass interpretation rule, while Figure 5b shows the exemplar rule. Note that the general form of the rule results in more incorrect region hypothesis (which could be filtered by constraints from the knowledge network). Although the examplar rule misses some grass regions, those found have high confidence.



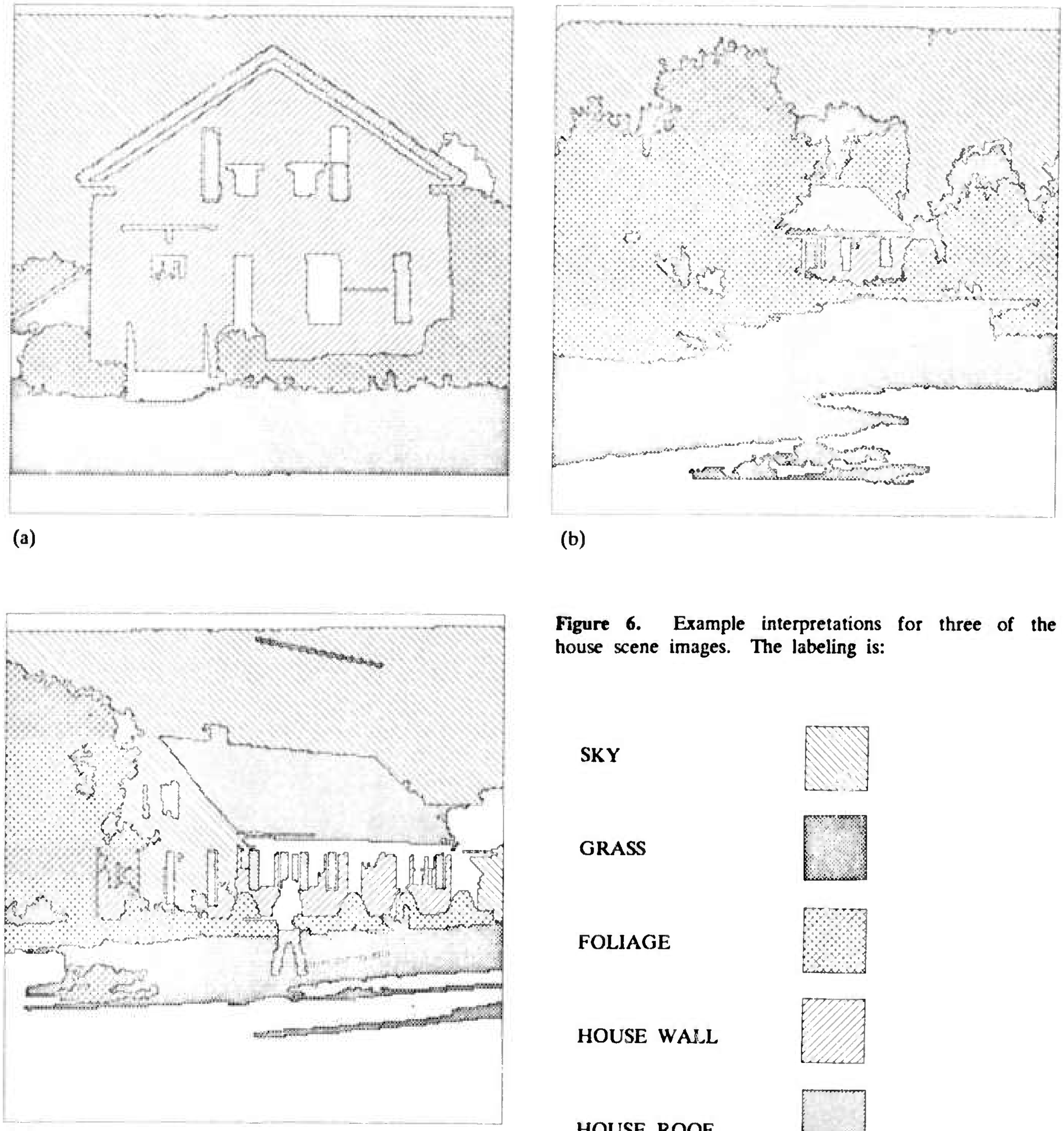


Figure 6. Example interpretations for three of the house scene images. The labeling is:

SKY	▧
GRASS	▓
FOLIAGE	░
HOUSE WALL	▨
HOUSE ROOF	▒
SHUTTER/WINDOW	░
UNLABELED	□

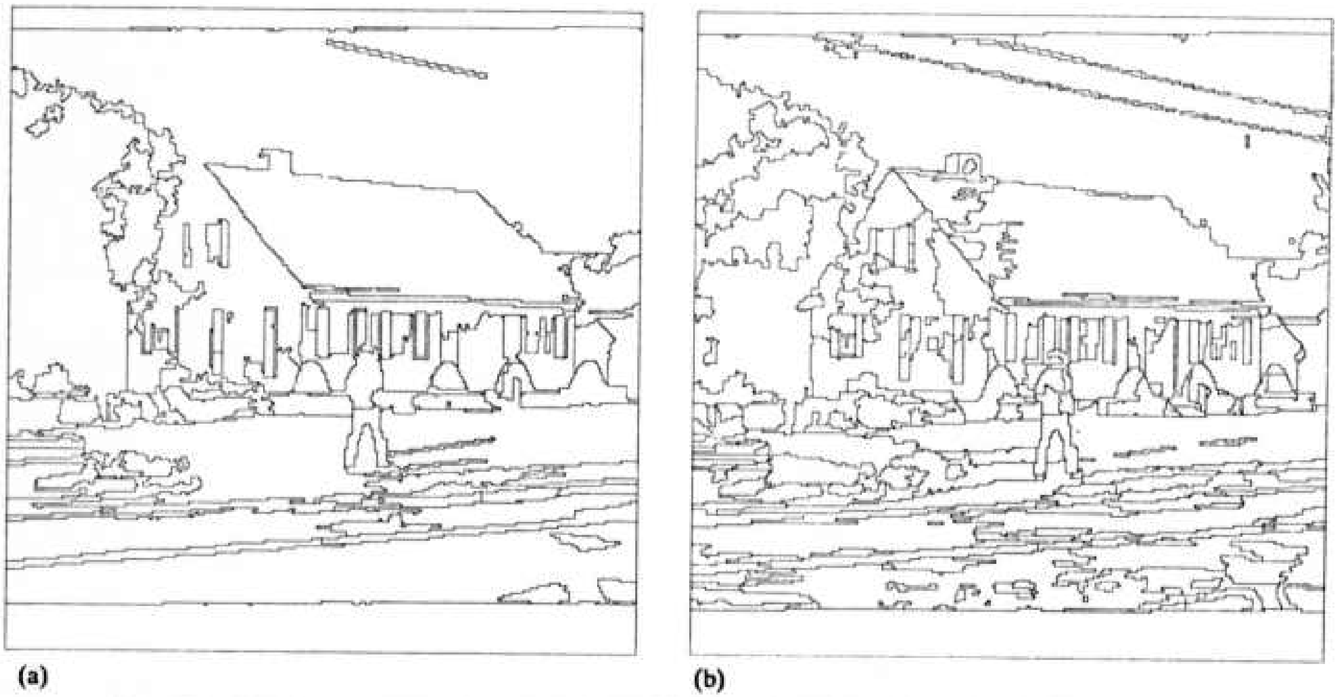


Figure 7. The Segmentation process can be varied to produce finer distinctions in image structure at the expense of a larger number of regions and subsequent fragmentation of many large regions. Figure 7a (course detail segmentation) exhibits missing roof/sky boundary; the finer detail segmentation (Figure 7b) has this boundary, although it was formed at the expense of significantly more regions.

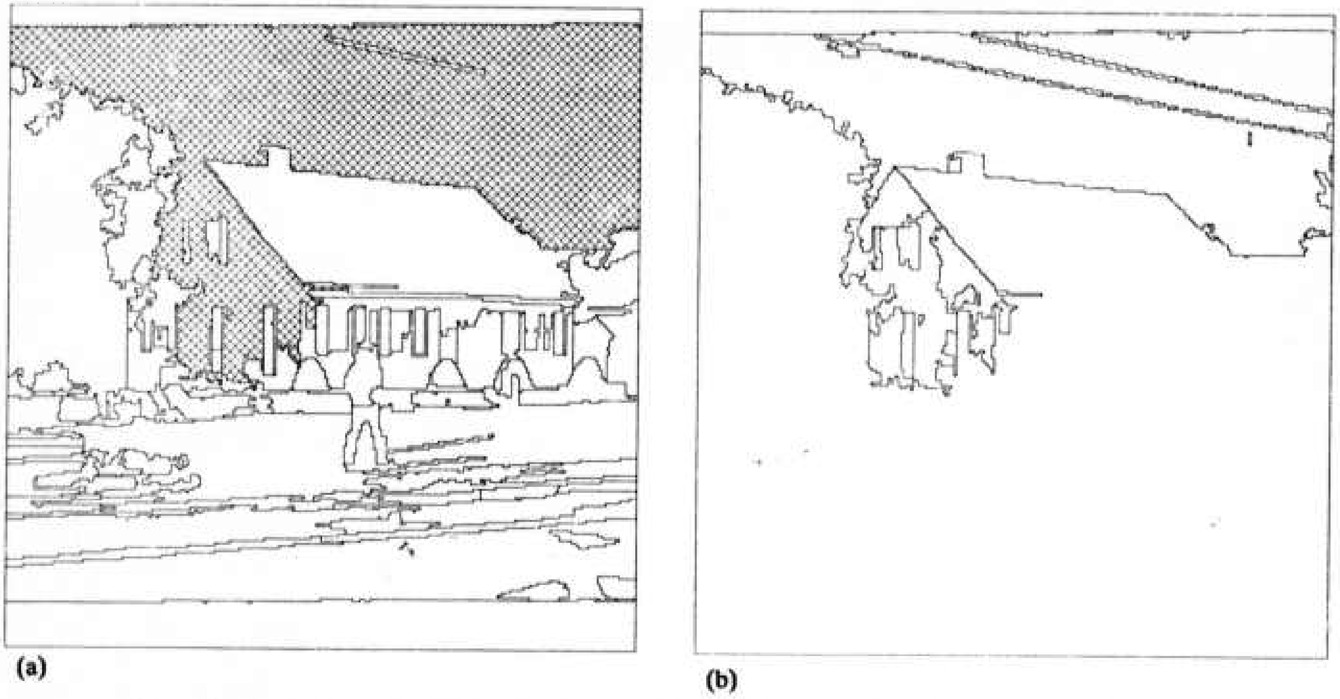


Figure 8. Resegmentation of house/sky region from Figure 7a. Figure 8a is the original segmentation showing the region to be resegmentated; 8b shows the regions resulting from the selective application to the segmentation process to the cross-hatched area in 8a.

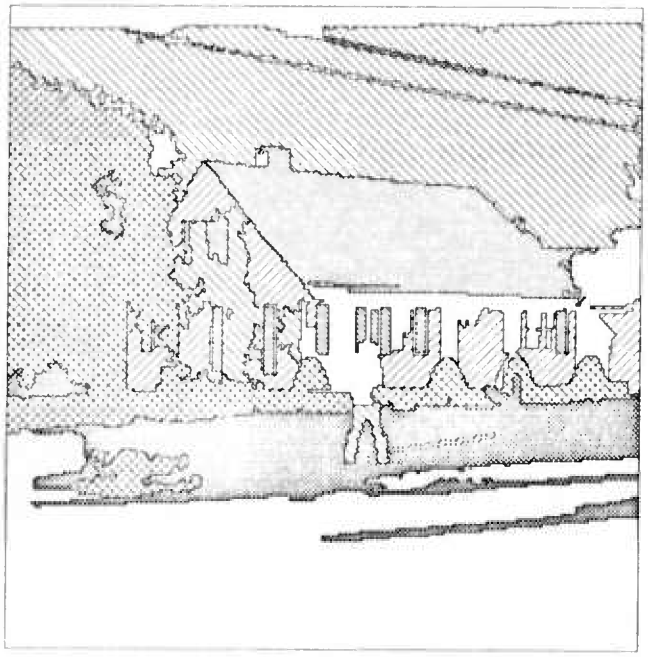
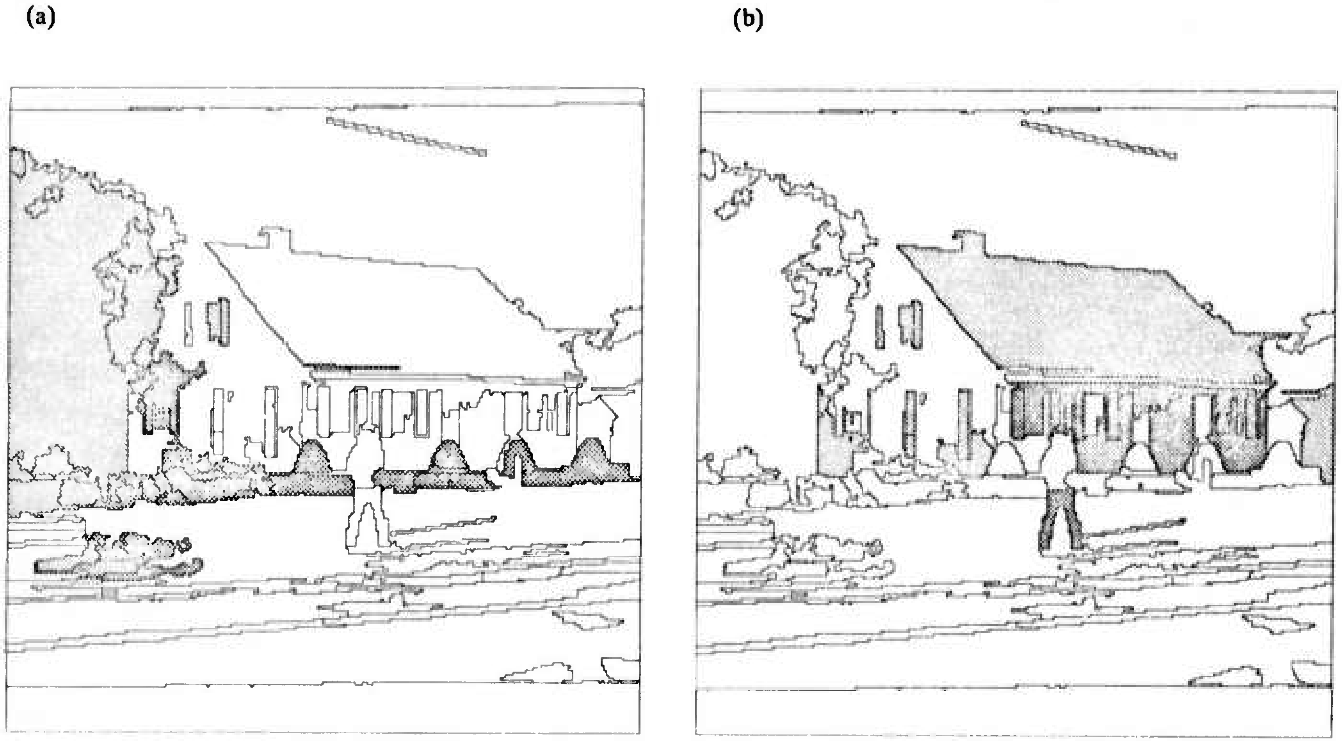


Figure 9. Final interpretation of the house scene in Figure 6c, after inserting resegmented houes/sky regions and reinterpreting the image.

Figure 10. An example of the use of spatial relations to filter and extend region labeling. The geometric relations between house and shrub (in 10a) and between between roof and house front wall (in 10b) are used to refine region hypotheses from the interpretation shown in Figure 6c. Note that there are still ambiguities (the shrub label in the grass area, and the pants labeled as house wall) that require the use of other filters.



THE PERCEPTUAL ORGANIZATION OF VISUAL IMAGES: SEGMENTATION AS A BASIS FOR RECOGNITION

David G. Lowe and Thomas O. Binford

Computer Science Department
Stanford University, Stanford, California 94305

Abstract

Evidence is presented showing that bottom-up grouping of image features is usually prerequisite to the recognition and interpretation of images. We describe three functions of these groupings: 1) segmentation, 2) three-dimensional interpretation, and 3) stable descriptions for accessing object models. Several unifying principles are hypothesized for determining which image relations should be formed: relations are significant to the extent that they are unlikely to have arisen by accident from the surrounding distribution of features, relations can only be formed where there are few alternatives within the same proximity, and relations must be based on properties which are invariant over a range of imaging conditions. Using these principles we develop an algorithm for curve segmentation which detects significant structure at multiple resolutions, including the linking of segments on the basis of curvilinearity. The algorithm is able to detect structures which no single-resolution algorithm could detect. Its performance is demonstrated on synthetic and natural image data.

Introduction

A major goal of computer vision research is to relate visual images to prior knowledge of their constituents, and thereby label and interpret them. However, current model-based vision systems have been demonstrated only in tightly-constrained environments with a few well-specified models to compare to the image [2, 9, 12]. The difficulty in expanding performance to more general domains is not one of ambiguity—it is very unlikely that two different models will fully fit the same image data. Rather, the problem is one of searching for potential correspondences between models and the image, since increasing the number and generality of the models results in an excessively large space of possible matches. Continued research into recovering three-dimensional shape from images—using stereo, motion, shading, and texture—promises to reduce the size of this search space considerably. However, the problem of matching is far from solved even when given full three-dimensional information, and these methods fail to explain the excellent level of human performance in such simple domains as line drawings.

In order to interpret images about which we have little prior knowledge, it is necessary to use effective bottom-up techniques to structure and describe the image in a form that can be used to selectively index into a large body of world knowledge. In this paper we will describe methods for detecting and evaluating the significance of relations between image elements in a way that can be applied uniformly to all images before we have any knowledge of their contents. Previous research on this and related topics has gone under such names as image segmentation, figure/ground phenomena, texture description, perceptual organization, and Gestalt perception. There have been many efforts to develop algorithms for specific segmentation problems, such as the detection of collinearity or connectivity, but these have not been integrated and have often lacked general applicability. Marr's initial primal sketch formulation [8] was intended to make some of these relations explicit. Recently, Witkin and Tenenbaum [13] have argued for the importance of detecting regularities and imposing structure on the image for many of the same reasons given here. They describe a unified treatment of inference based on the assumption that regularities detected in the image are non-accidental. In this paper we will describe the role that this form of inference plays in model-based recognition, develop some underlying principles for this level of interpretation, and present new segmentation methods based upon these principles.

There are three valuable sources of information which the bottom-up organization of image features can provide, all of which simplify the problem of matching against world knowledge:

1) A major reduction in the search space is achieved by segmentation—the division of the image into sets of related features. This has long been recognized as a crucial problem in image interpretation. We do not want to match models against all possible combinations of features in an image, so good segmentation is crucial for reducing the combinatorics of this search.
2) Two-dimensional relations lead to specific three-dimensional interpretations, as we have described in previous papers [1, 5]. For example, collinear lines in the image are likely to be collinear in 3-space. A corollary of this is that these image relations tend to be invariant

with respect to viewpoint, which greatly simplifies the problem of matching to three-dimensional objects of unknown orientation.

3) To the extent that these relations are stable under different imaging conditions and viewpoints, they can be used as index terms to access a body of world knowledge. Not only can the names of the relations be used, but in addition each relation will have several parameters of variation whose relative values in the image can be used. For example, collinear line segments can be characterized by the relative sizes of the segments and gaps, which provides a viewpoint-invariant description that can be used to select a model for attempted matching.

Note that all three of these points assume that the relations found in the image are a result of regularities in the objects being viewed, which means that any relations which happen to arise accidentally from independent features will only confuse the interpretations. This distinction between significant and accidental relations is a point to which we will return.

The importance of segmentation for recognition: An experiment

The importance of these grouping operations as a stage in the processing of images by the human visual system can be demonstrated by a straightforward psychophysical experiment. In Figure 1(a) we have constructed a partial line drawing of a bicycle in such a way that most opportunities for bottom-up segmentation are eliminated (e.g., we have eliminated most cases of significant collinearity, endpoint proximity, parallelism, and symmetry). In informal experiments with 10 subjects who were told nothing about the identity of the object, this drawing proved to be remarkably difficult to recognize. Nine out of 10 subjects were unable to recognize the object within a 60 second time limit, and the tenth subject took 45 seconds. Note that this is in spite of the fact that the object level segmentation has already been performed—the task would be even harder if the bicycle were embedded in a normal scene containing many surrounding features.

Figure 1(b) is the same drawing as in 1(a) with only a single segment added. The added segment was placed in a strategic location which would allow it to be combined with other segments in a curvilinear grouping. The center of this circular grouping would then be coincident with the termination of another segment, leading to further groupings. As might be expected if we assume that bottom-up groupings play an important role in recognition, the recognition times for this second figure were dramatically lower than for the first, with 3 out of 10 subjects recognizing it within 5 seconds and with 7 out of 10 subjects recognizing it within the 60 second time limit. Presumably, if the added segment had been placed at some location which did not lend itself to perceptual groupings the change in recognition times would have been negligible.

These figures can also be used to demonstrate the human capability to make use of top-down contextual information to limit the search space for forming a match. As was demonstrated in experiments performed as early as 1935 [3], verbal clues naming the object in an image or even naming vague non-visual object classes can drastically reduce the recognition time. Subjects can usually interpret Figure 1(a) immediately upon being told that it is a bicycle. Thus this figure is on an interesting borderline where either bottom-up or top-down information can suddenly reduce the search space and lead to recognition. One can imagine a series of experiments that would systematically explore this search space and the reduction in its size created by different bottom-up or top-down clues. These figures can also be used to demonstrate the human equivalent of a back-projection algorithm [4] followed by image-level matching, where certain hypothesized partial matches can be used to solve for the position, orientation, and internal parameters of the model, which in turn lead to predictions for further matches at specific locations in the image.

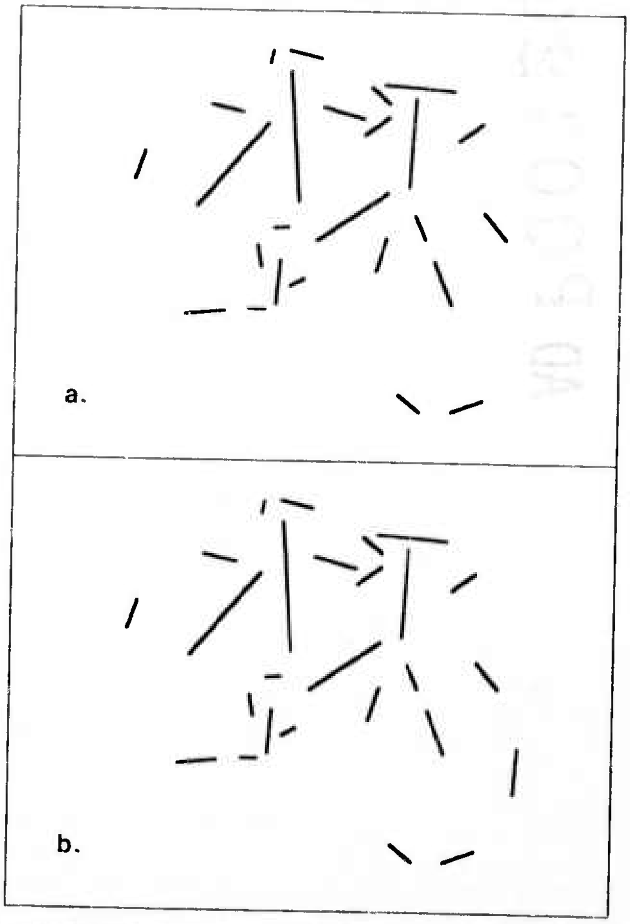


Figure 1: When opportunities for bottom-up grouping of image features have been removed, as was done for the line drawing of a bicycle in (a), the drawing is remarkably difficult to recognize. The average recognition time for (a) was over one minute when the subjects had no prior knowledge of the object's identity. When a single line segment was added in (b), which provided local evidence for a curvilinear grouping, the recognition times were greatly reduced.

Principles of segmentation

There are virtually an infinite number of relations that could be formed between the elements of any image. What general principles can we derive for selecting those relations which are worth forming and for measuring their significance? As was mentioned earlier, segmentations are useful only to the extent that they represent actual structure of the scene rather than accidental alignments. Therefore, a central function of the segmentation process must be to distinguish, as accurately as possible, significant structures from those which have arisen at random. All of the relations we have considered can arise from accidents of viewpoint or random positioning as well as from structure in the image. However, by examining the accuracy of each relation and the surrounding distribution of features in the image, it is possible to give probabilistic measures of the likelihood that any given relation is accidental. These nonrandomness measures can then be used as the basic test for significance during the segmentation process.

If there were a significant level of prior knowledge regarding the expected distributions of features and relations, this could be used for judging the significance of segmentations. However, the range of common images seems to be so wide that any prior knowledge at this level must be very weak. We have chosen to carry out our computation of significance with respect to the null hypothesis that features are independent with respect to orientation, location, and scale. Significance is then inversely proportional to the probability that the relation would have arisen from such a set of independent features. It is a matter for psychological experimentation to see whether the human visual system is biased in any direction from this independence assumption. But since a scene typically contains many independently positioned objects (leading to independence with respect to orientation, location, and scale in the image), the discrimination of relations with respect to this background seems like a reasonable criterion for judging significance.

A second major principle of segmentation is that each operation must have limited computational complexity. It is obviously impossible to test all combinations of features in an image, so the relations can only be formed over distances that do not include too many false candidates of the particular type being examined [6]. Figure 2 shows an example in which a highly significant grouping of five equally-spaced collinear dots is not apparent to human vision when there are enough surrounding false targets. It would presumably be useful for the purposes of interpretation and recognition to detect such a statistically significant grouping, so this failure must be attributable to a lack of computational resources. This does not mean that groupings are diameter-limited in any absolute sense, since groupings can be attempted at many different scales; however, if there are more than a few false candidates at some scale, then no groupings can be formed at that scale of description.

The principles above describe which groupings will be formed and how they will be evaluated for a given class of relations, but they do not specify which classes of relations will be attempted. There are several factors which influence this choice. One important factor is the same imaging-invariance condition that was mentioned earlier—it is only worth looking for image relations which do not depend on a specific viewpoint, light-source position, or other image-formation parameter. For example, collinearity is useful because it is present in the image over all viewpoints of collinearity in the scene. But it would be pointless to detect lines at right-angles in the image, since even if right-angles are common in the scene the angle in the image would change with almost any change in viewpoint. More generally, Witkin and Tenenbaum [13] argue that prior probabilities play a role in selecting which relations are the easiest to distinguish from accidentals, and should therefore be attempted. If some relation arises only very rarely from the structure of typical scenes, then it is more likly that some instance of the relation in an image is accidental (although it would still be possible to distinguish the relation from accidentals given accurate-enough image measurements). Of course, it is also less productive to spend time searching for properties which seldom arise than for those which are common.

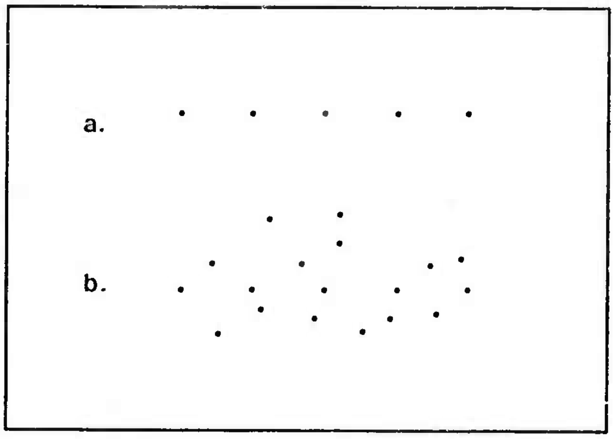


Figure 2: The pattern of five equally-spaced collinear dots in (a) is not detected spontaneously by human vision if it is surrounded by enough competing candidates for grouping, as in (b). This occurs even though the relation remains highly significant in the statistical sense and would therefore likely be of use for recognition.

An algorithm for curve segmentation

A significant bottleneck in creating a computer program which can perform these bottom-up perceptual processes on natural images is the problem of creating appropriately segmented edge desriptions. The best current edge operators detect "edge points" which are then linked using nearest-neighbor algorithms into lists of points. Although there has been considerable research into the problem of fitting smooth curves to these lists of points [10, 11, 12], almost without exception these efforts have concentrated on a single pre-selected resolution of segmentation and have attempted merely to smooth out noise induced by the imaging process. Although these smoothed results may appear reasonable to the naïve human eye, that is because the human visual system can still perform the lower resolution groupings even though they have not been detected and described by the program. Figure 3 illustrates the problem, where the segmentation in Figure 3(b) is adequate to recognize one instance of collinearity, but other groupings are only apparent when lower resolution structures are recognized as in Figure 3(c). It is not adequate to merely apply previous single-resolution methods at multiple resolutions, since only some structures at some resolutions will be significant and the measurement of this significance is important for further inference. Therefore, we have developed a new algorithm, based on the principles of segmentation outlined earlier, which measures the degree of nonrandom structure in edge-point lists over a wide range of resolutions. In our implementation, we examine all groupings which are either linear or of constant curvature. These can be splined to represent arbitrary smooth curves, although it is possible that human vision includes the detection of more general primitive curve groupings, such as spirals.

Measuring the significance of a curve segmentation

The first task in developing a segmentation algorithm is to determine how we will measure the significance of each grouping. In this case, since the points were originally linked on the basis of proximity, we must be careful not to confuse nonrandomness in proximity with the measurement of nonrandomness in linearity. For example, if we start by looking at a set of only three points, we might measure the significance of their linearity by measuring the distance of one point from the line joining the other two. However, this would confuse the effects of proximity with those of linearity, since by being close to one of the other points the third point would automatically be close to the line on which they lie, as is shown in Figures 4(a) and 4(b). Therefore, we have chosen to define nonrandomness in linearity to be how unlikely a point is to be as close as it is to a curve given its distance from the closest defining point of the curve. This is equal to $2\theta/\pi$, where θ is the angle between the line and the vector from the closest endpoint, which for points close to the line is approximately equal to the distance from the line divided by the distance from the closest endpoint. This can be extended to 4 or more points by recursively looking for the point which is farthest away from any of the points considered thus far and calculating the likelihood for that point in terms of its minimum proximity to these previously considered points. Since these likelihood values are independent, they can be multiplied together to produce an overall value for the curve.

The algorithm could be made symmetric with respect to the set of points by using some sort of best-fit curve rather than selecting a subset of points to define the curve. However, for the results displayed in this paper we have adopted the computationally expedient but less accurate method of defining the line by the two points with greatest separation and adding the most central point to define a circular arc. Note that this test will only be used to measure significance and not the final positions of curve segments,

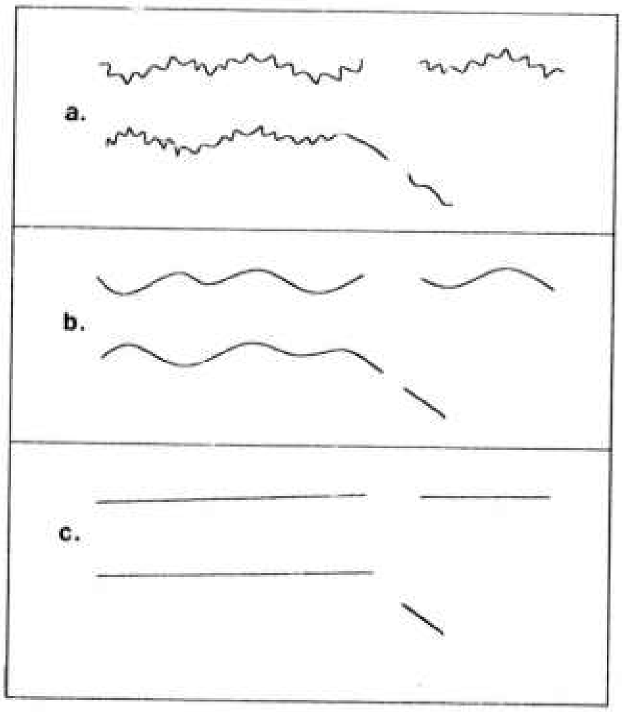


Figure 3: The data in (a) can be segmented at at least two different resolutions of description, as shown in (b) and (c). One instance of collinearity can only be detected in segmentation (b) while the other instance of collinearity and the parallelism can only be detected in (c).

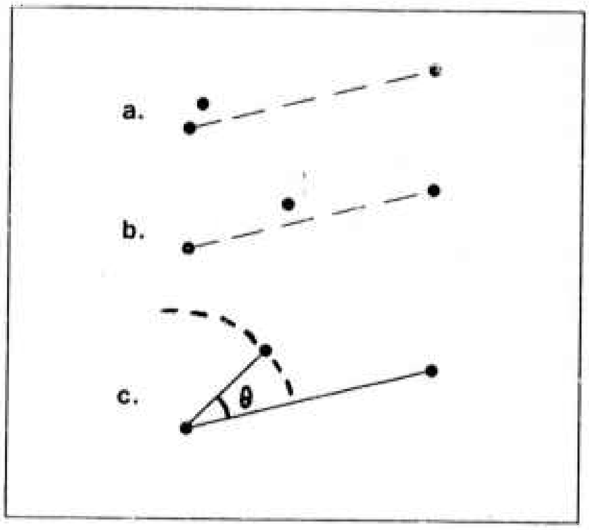


Figure 4: The middle dots in (a) and (b) are both the same distance from the lines joining the other two dots. Yet the three dots in (b) are much more significant in terms of their collinearity than those in (a), since the middle dot in (a) could be close to the line merely as a result of its proximity to the first endpoint. Therefore, we measure the probability of a point being within a given distance from a line in terms of its proximity to the closest endpoint defining the line, as shown in (c).

and that small relative errors in ranking segmentations will not be very important—our nonrandomness values (inverse of probability) range from a minimum of 1.0 for apparently random points up to at least 10^6 for many points placed accurately along smooth curves.

Testing all possible groupings

Given this significance test for sets of points, we want to divide the initial linked list of points into segments which lead to the highest significance values. Previous methods of curve segmentation have usually attempted to search for corners (tangent discontinuities) on a curve, where the curve can be divided into different segments. Our approach is the dual—we look for segments of the curve which exhibit significant nonrandomness, and tangent and curvature discontinuities are assigned to the junctions between neighboring segments. In contrast to the earlier approaches, our method will fail to assign any segmentation where the curve appears to wander randomly at all resolutions, and will assign multiple segmentations where it exhibits different structure at different resolutions.

It would clearly be too costly to test every possible segment of the curve for nonrandomness. However, if we allow a reasonable margin of error, it is possible to cover all scales and locations with a relatively small number of groupings. We have chosen to examine groupings at all scales differing by factors of two, from groupings of only three adjacent points up to groupings the size of the full length of the curve (amounting to 6 scales for a curve of 100 points). At each scale, we examine groupings at all locations along the curve, with adjacent groupings overlapping by 50%. This means that any given segment of the curve will have at least one grouping attempted which covers 50% of its length but does not extend outside its borders. In addition, while calculating the nonrandomness measure, each segment is extended to include points on both sides which are close enough to the curve that their inclusion increases the nonrandomness value.

Many of the segments produced by this exhaustive testing will not exhibit significant nonrandomness and others will be qualitatively the same as larger segments of which they are merely a subset. Therefore, a thinning procedure is executed which steps through the different resolutions at each location along the curve and selects only those segmentations which are locally maximum in their significance values. It is possible that there will be more than one local maximum if the curve exhibits different structures at different resolutions of grouping. There is also a threshold at the 0.05 significance level, below which groupings are not considered significant. Once again, these choices are somewhat expedient, and we are seeking a more fundamental method of combining multiple resolutions.

The algorithm in action

This algorithm have been implemented in MACLISP on a KL-10 computer and tested on synthetic data as well as edges derived from natural images. Figure 5(a) shows some hand-drawn curves which exhibit different structures at different resolutions, much as was shown in Figure 3. Figure 5(b) gives the output of the curve segmentation algorithm when given this data, and demonstrates the algorithm's ability to detect significant structure at multiple resolutions—results which no single-resolution algorithm could have produced.

Figure 6 shows the results of running the algorithm on a small 30 by 45 pixel region of an aerial photograph of an oil tank facility. The original digitized image is shown in 6(a). Figure 6(b) shows some linked edge data generated from this image by an edge detection program written by David Marimont [7], which detects edge points to subpixel accuracy and links them into lists. Figure 6(c) shows all the groupings at all resolutions, although the widely differing significance values are not apparent. Figure 6(d) shows the results after the thinning process which selects local maxima with respect to resolution. Given these segments, it is relatively easy to form collinearity and curvilinearity relations between them as shown by the dotted lines in Figure 6(e). It would also be fairly straightforward to detect endpoint proximity, parallelism, constant intervals, and other perceptual groupings.

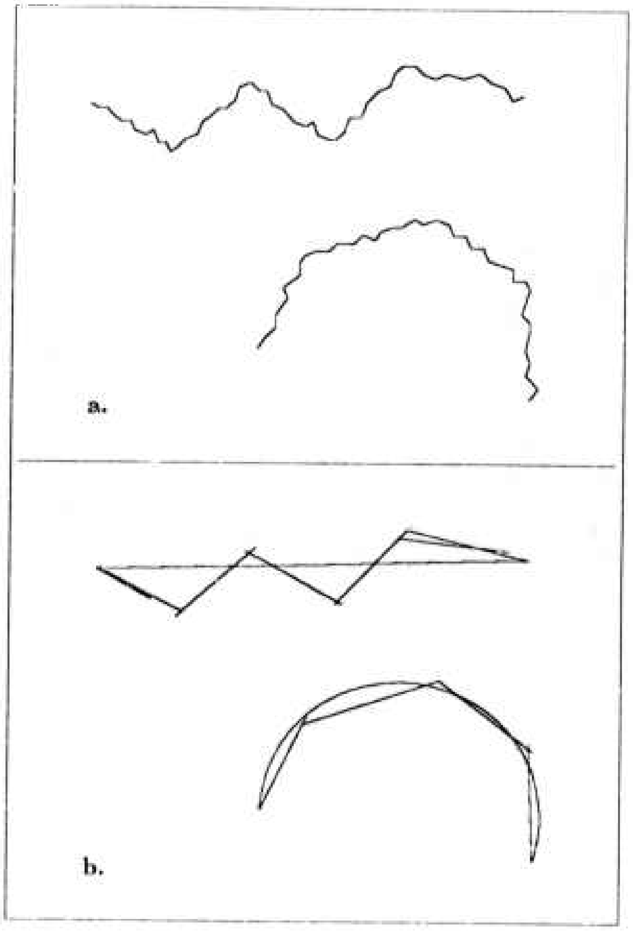
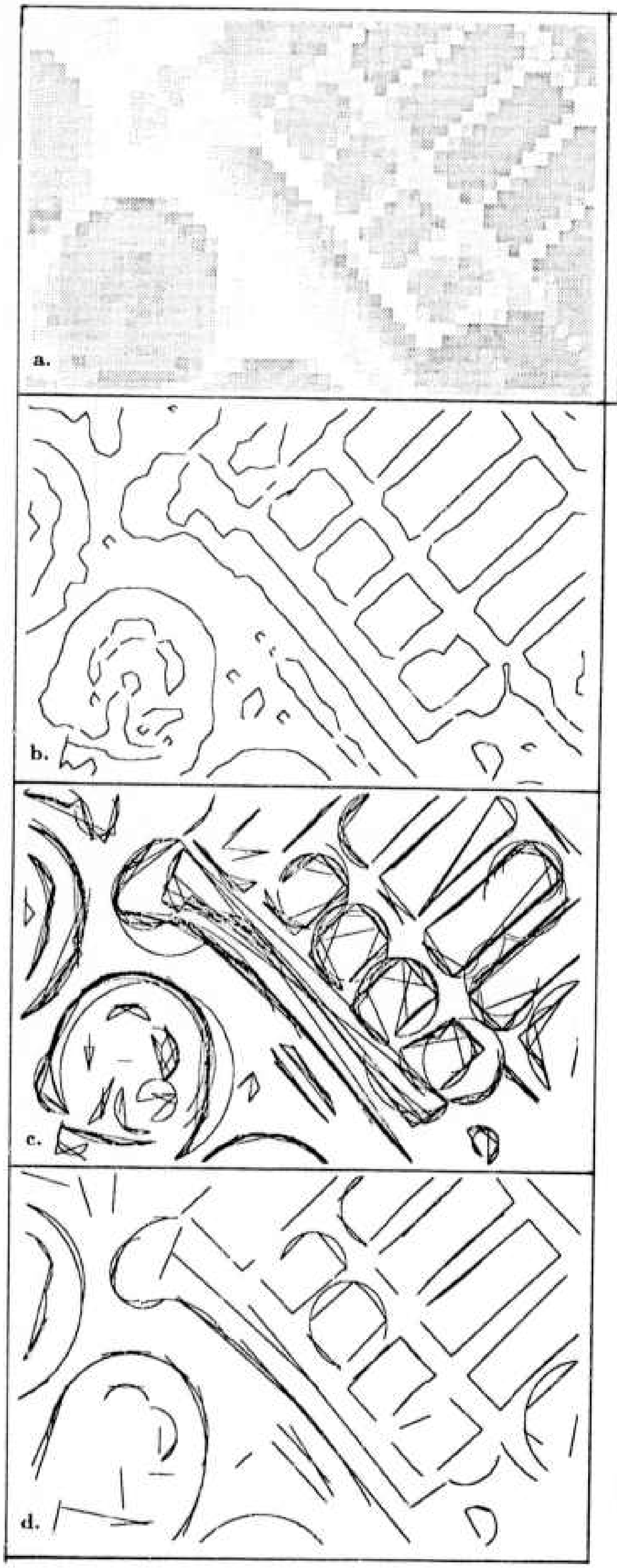


Figure 5: The hand-input curves in (a) have been created to exhibit significant structure at multiple resolutions. When these are given as data to the curve segmentation algorithm, it produces the results shown in (b), which makes these multiple levels of structure explicit.



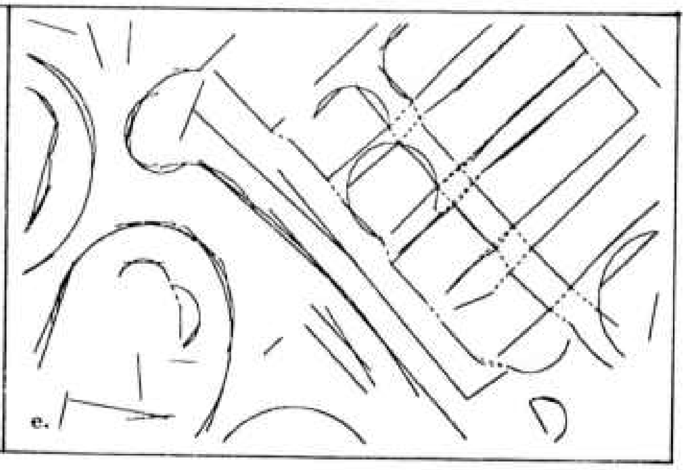


Figure 6: The small 30 by 45 pixel region of an aerial photograph shown in (a) was run through the Marimont edge detector to produce the linked edge points shown in (b). Figure (c) shows all the segments at different scales and locations which were tested for significance. After selecting only those segments which were locally maximum with respect to size of grouping, and thresholding out those which are not statistically significant, we are left with the segments shown in (d). It is then fairly simple to form collinearity and curvilinearity relations between these segments as shown by dotted lines in (e).

Summary

We started this paper by demonstrating the importance of bottom-up perceptual organization for human vision. These image relations play a major role in limiting the size of the search space that must be considered when matching against world knowledge. The unifying principles of detecting non-random structure, avoiding combinatorial complexity, and looking for viewpoint-invariant relations were suggested. An algorithm for curve segmentation, based upon these principles, was developed and demonstrated. These curve segmentations enabled the use of a relatively simple algorithm for grouping on the basis of curvilinearity, and extensions for detecting other classes of groupings seem to be within reach. There are many other problems besides recognition in which these groupings would be useful. An example is the stereo correspondence problem, since to the extent that these image relations represent structure in the scene and are invariant with respect to viewpoint, they can be expected to remain visible in images taken from different viewpoints. They would then provide far less ambiguous structure for matching than simple edge points.

The specific algorithms developed are preliminary implementations of the general methodology of segmenting perceptual data by looking at groupings over a wide range of scales and locations and retaining those which are the most unlikely to have arisen by accident from the background distribution. This same methodology could be applied to a wide range of other perceptual segmentation problems or signal analysis.

Acknowledgements

We would like to thank Peter Blicher, Chris Goad, David Marimont, Marty Tenenbaum, Brian Wandell, Andrew Witkin, and our other colleagues for stimulating discussions and suggestions. This work was supported under ARPA contract N00039-82-C-0250. The first author was also supported by a postgraduate scholarship from the National Research Council of Canada.

References

[1] Binford, Thomas O., "Inferring surfaces from images," *Artificial Intelligence,* **17** (1981), 205-244.

[2] Brooks, Rodney A., "Symbolic reasoning among 3-D models and 2-D images," *Artificial Intelligence,* **16** (1981).

[3] Leeper, R., "A study of a neglected portion of learning—the development of sensory organization," *Journal of Genetic Psychology,* **46** (1935), 41-75.

[4] Lowe, David G., "Solving for the parameters of object models from image descriptions" *Proceedings ARPA Image Understanding Workshop* (College Park, MD, April 1980), 121-127.

[5] Lowe, David G. and Thomas O. Binford, "The interpretation of three-dimensional structure from image curves," *Proceedings IJCAI-7,* (Vancouver, Canada, August 1979), 613-618.

[6] Lowe, David G. and Thomas O. Binford, "Segmentation and aggregation: An approach to figure-ground phenomena," *Proceedings ARPA Image Understanding Workshop* (Stanford, CA, Sept. 1982).

[7] Marimont, David, "Segmentation in ACRONYM," *Proceedings ARPA Image Understanding Workshop* (Stanford, California, September 1982).

[8] Marr, David, "Early processing of visual information," *Philosophical Transactions of the Royal Society of London, Series B,* **275** (1976), 483-524.

[9] Nevatia, R., and T.O. Binford, "Description and recognition of curved objects," *Artificial Intelligence,* **9** (1977).

[10] Pavlidis, T., *Structural Pattern Recognition* (New York: Springer-Verlag, 1977).

[11] Rutkowski, W.S., and Azriel Rosenfeld, "A comparison of corner-detection techniques for chain-coded curves," TR-623, Computer Science Center, University of Maryland, 1978.

[12] Shirai, Y., "Recognition of man-made objects using edge cues," *Computer Vision Systems,* A. Hanson, E. Riseman, eds. (New York: Academic Press, 1978).

[13] Witkin, Andrew P. and Jay M. Tenenbaum, "On the role of structure in vision." To appear in *Human and Machine Vision,* Rosenfeld and Beck, eds. (New York: Academic Press, 1983).

The Theory of Straight Homogeneous Generalized Cylinders

Steven A. Shafer and Takeo Kanade

Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA

Abstract

In recent years, Binford's generalized cylinders have become an important tool for image understanding. However, research has been hampered by a lack of analytical results for these shapes. In this paper, a definition is presented for Straight Homogeneous Generalized Cylinders, those generalized cylinders, with a straight axis and with cross-sections which have constant shape but vary in size. This class of shapes, while still quite large, has properties which make considerable analysis possible.

The results begin with deriving formulae for points and surface normals for these shapes. Theorems are presented concerning the conditions under which multiple descriptions can exist for a single solid shape. Then projections and contour generators are analyzed for some subclasses of shapes. The strongest results are obtained for solids of revolution (which we name Right Circular SHGCs), for which a closed-form method for analyzing image contours is presented. It is seen that a picture of the contours of a solid of revolution is ambiguous, with one degree of freedom related to the angle between the line of sight and the solid's axis.

1. Introduction

In recent years, the *generalized cylinders* proposed by Binford [2] have become a popular shape representation scheme for image understanding. Unfortunately, research has been hampered by a lack of analytical results for these shapes. This paper presents Straight Homogeneous Generalized Cylinders and performs analysis of several of their basic properties.

Straight Homogeneous Generalized Cylinders (SHGCs) are defined by a cross-section shape which sweeps along a straight line axis, changing size uniformly as it is swept. This class of shapes, while still quite large in itself, has properties which allow considerable analysis to be performed. Formulae are developed for points on the surface of an SHGC and for surface normals of an SHGC.

Many researchers in the past have approached generalized cylinders by trying to specify the "canonical" description for a given shape. We take a different approach, allowing a shape to have many different descriptions as a generalized cylinder. This solves the very difficult problem of trying to specify a single description as the "best"; however, it introduces the problem of deciding when two descriptions are in fact describing the same shape. This "Equivalence Problem" is given some attention and some limited (but still very useful) results are presented.

We then describe contours of tangency with the line of sight, which will include "outlines" and visible "folds" in an image of an SHGC. We examine in some detail the special case of Right Circular SHGCs (solids of revolution), which have sufficiently strong properties to allow very detailed contour analysis. Using these shapes, we develop a contour analysis technique and make several interesting observations about occlusion and singularities in tangency contours. We show that an image of the contours of a solid of revolution is ambiguous with one degree of freedom in the interpretation.

2. Straight Homogeneous Generalized Cylinders

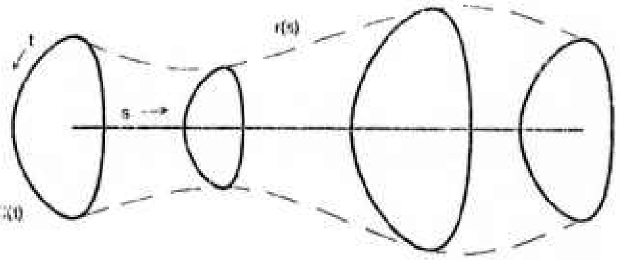


Figure 1: Straight Homogeneous Generalized Cylinder

Figure 1 shows a *Straight Homogeneous Generalized Cylinder (SHGC)*, as described in the taxonomy of [8]. An SHGC is a function which maps two parameters onto a set of points in x-y-z space (i.e. the world). The two parameters are s, which measures distance along the axis, and t, which indirectly measures distance along the cross-section; both s and t have as domain the unit interval [0,1]. This development is similar to that of Ballard and Brown [1].

A Straight Homogeneous Generalized Cylinder is specified by a four-tuple ($\boldsymbol{A}$, $\boldsymbol{C}$, r, α). $\boldsymbol{A}$ is the *axis*, which is a curve in space defined in parametric form by $A(s) = (x_A, y_A, z_A)(s)$. The remainder of this discussion will desribe features of the shape relative to the axis itself rather than in absolute x-y-z coordinates.

At each point $A(s)$ on the axis, let the cross-section be described on a u-v plane, with $A(s)$ at the origin, and defined by the (constant) angle α. The u-axis will be the direction of steepest descent of the u-v plane from the tangent to the axis (i.e. the projection of the tangent to the axis onto the u-v plane). α, the *angle of inclination*, is the angle from the u-axis to the tangent to the axis at $A(s)$; $\alpha = 0$ means that the u-axis is pointing towards $A(1)$, and $\alpha = \pi$ means that the u-axis is pointing towards $A(0)$.

In a Straight Homogeneous Generalized Cylinder, the "Straight" attribute implies that $\mathbf{A}(s)$ is a linear function. The axis is thus a line segment, and all u-v planes are parallel. This property is important because all tangents to $\mathbf{A}(s)$ are parallel, as are all u-axes and all v-axes. We can therefore define vectors $\mathbf{S}$, $\mathbf{U}$, and $\mathbf{V}$ pointing in these directions and assign a local (object-centered) coordinate system using u-v-s coordinates. Such coordinates can of course be defined for all Generalized Cylinders; however, there will be no curvature in the coordinate axes for SGCs. The local coordinates of an SGC are a linear transformation of world (x-y-z) coordinates.

On the u-v cross-section plane for each value of s, the cross-section is the set of points $r(s)\mathbf{C}(t)$ for values of t from 0 to 1, inclusive. The *cross-section function* $\mathbf{C}(t) = (u_C, v_C)(t)$ describes the *shape* of the cross-section; the *radius function* $r(s)$ describes its *size*. So, the cross-sections have the same shape but may vary in size; this property is implied by the "Homogeneous" attribute. The union of the cross-sections is the surface of the Straight Homogeneous Generalized Cylinder.

According to this strict definition, some very peculiar shapes are allowed as SHGCs, including those with singular points or arcs and those with cross-sections that are open arcs, points, or even space-filling curves. Since our ultimate goal is the analysis of the shapes of common objects, we will generally exclude such bizarre cases from further consideration. However, since shapes with degenerate cross-sections (open arcs or points) do have several important properties, we will note them when appropriate.

We impose the restriction that the functions A and r be continuous and differentiable everywhere and that the cross-section $\mathbf{C}$ be continuous and differentiable almost everywhere. It is usual, but not required, that the u-v origin be in the interior of the cross-section. In addition, we will presume "uniform scaling" of s and t, i.e. $||d\mathbf{A}/ds||$ and $||d\mathbf{C}/dt||$ are constants.

2.1 Subclasses of SHGC

We will also be referring to several subclasses of SHGC with particular interesting properties:

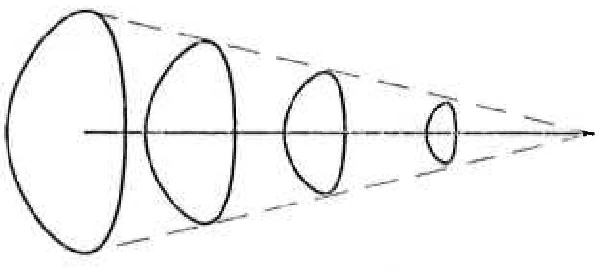


Figure 2: Linear SHGC

Linear SHGC (LSHGC) -- SHGC with r linear (figure 2)

The size of the cross-section varies linearly with distance along the axis. LSHGCs are *ruled surfaces* as well as being Generalized Cylinders [4].

Right SHGC (RSHGC) -- SHGC with $\alpha = \pi/2$

The u-v planes are normal to the axis. There is no "direction of steepest descent" relative to the axis, so the u-axis may be chosen in any direction on the cross-section planes.

Circular SHGC (CSHGC) -- SHGC with $\mathbf{C}$ a circle centered at the origin

Without loss of generality, let $\mathbf{C}$ be a unit circle, $\mathbf{C}(t) = (u_C, v_C)(t) = (\cos 2\pi t, \sin 2\pi t)$. All surfaces of solids of revolution are Right Circular SHGCs (but with open ends unless $r(0) = 0$ or $r(1) = 0$).

Polygonal SHGC (PSHGC) -- SHGC with $\mathbf{C}$ polygonal (piecewise linear)

If $\mathbf{C}(t_0)$ is a vertex for some t_0, then the set of points $\mathbf{P}(s,t_0)$ is a *crease* (ridge if $\mathbf{C}$ convex there, valley if concave). Otherwise, $\mathbf{P}(s,t)$ is on a *face*; note that faces are not necessarily planar in this definition.

In various situations, the consequences of these properties will be shown to be of special interest.

3. Coordinates for SHGCs

For any SHGC, there is a natural u-v-s object-centered coordinate system imposed by the preceding definition.

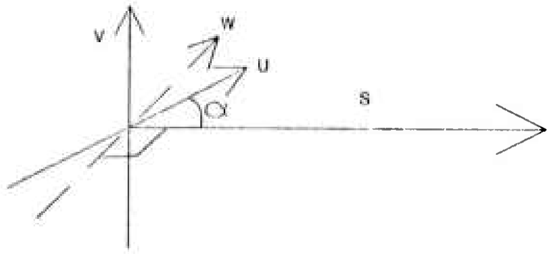


Figure 3: Coordinate Axes for SHGCs

We will adopt the convention that the v-axis is chosen to provide a right-handed u-v-s coordinate system. The unit vectors in the axis directions will be denoted $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{S}$, as shown in figure 3. It will be convenient to define an orthogonal w-v-s coordinate system using $\mathbf{W}$ perpendicular to $\mathbf{V}$ and $\mathbf{S}$. For any point $(u, v, s)_{uvs}$ (where $_{uvs}$ denotes coordinates in the u-v-s system), the corresponding coordinates in w-v-s are $(u \sin\alpha,\ v,\ s + u \cos\alpha)_{wvs}$. In a Right SHGC, since $\mathbf{U} = \mathbf{W}$, u-v-s and w-v-s coordinates are identical.

We will use the notation $_{wvs}$ (etc.) whenever the coordinates are given in a system other than the world (x-y-z) or image (x-y) systems.

3.1 Points on the Surface and Surface Normals

For any values s and t, the point $\mathbf{P}(s, t)$ on the surface of the SHGC has w-v-s coordinates:

$$\mathbf{P}(s,t) = (u_C(t)\, r(s) \sin\alpha,\ v_C(t)\, r(s),\ s + u_C(t)\, r(s) \cos\alpha)_{wvs} \tag{3-1}$$

Surface normals for an SHGC can be defined wherever the cross-section function $\mathbf{C}(t)$ is differentiable. The outward-pointing surface normal vector $\mathbf{N}(s,t)$ at $\mathbf{P}(s,t)$ is the cross product of the tangent vectors to the surface in the directions of increasing s and t:

$$\mathbf{N}(s,t) = \frac{\partial \mathbf{P}}{\partial t} \mathrm{X} \frac{\partial \mathbf{P}}{\partial s}$$

We will use $N(s,t)$ parallel to $N'(s,t)$, defined by

$$N(s.t) = \frac{N'(s,t)}{r(s)}$$

$$= \left(h(t) \cos \alpha \frac{dr}{ds} + \frac{dv_C}{dt}, - \sin \alpha \frac{du_C}{dt}, - h(t) \sin \alpha \frac{dr}{ds}\right)_{\mathbf{wvs}}$$

where $h(t)$, defined by

$$h(t) = u_C(t) \frac{dv_C}{dt} - v_C(t) \frac{du_C}{dt} = \begin{vmatrix} u_C & v_C \\ du_C/dt & dv_C/dt \end{vmatrix}$$

is the *Wronskian* of the cross-section functions u_C and v_C [7]; $h(t) = 0$ implies that the SHGC has a line segment for a cross-section, i.e. is degenerate.

The surface normals of an SHGC obey two very important properties, stated in the *Corresponding Normal Theorem*:

> A non-degenerate SHGC is Linear iff, for all s, $\partial N/\partial s = (0,0,0)$; a non-degenerate SHGC is Polygonal iff for almost all t, $\partial N/\partial t = (0,0,0)$.

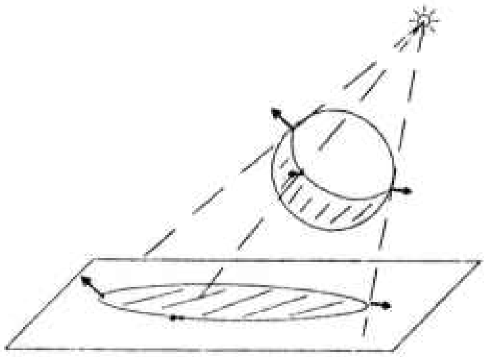


Figure 4: Surface Normals of the Shadow Volume

This says that the surface normals for an LSHGC depend only on t (i.e. are parallel along contours of constant t), and for a face of a PSHGC depend only on s (i.e. are parallel along contours of constant s within a face). A proof of the Corresponding Normal Theorem is presented in [8]. The Corresponding Normal Theorem is especially useful in shadow geometry, since the "shadow volume" (the volume of space shaded by an object) is an LSHGC (figure 4).

4. The Equivalence Problem for Shape Descriptions

A subtle problem arises from our definition of SHGCs: as we have defined SHGCs, an SHGC is actually a *description of a shape* rather than being a specific solid shape itself. Of course, each such description describes a unique shape; however, we must attempt to decide when a single shape may have several different descriptions. Since a solid shape corresponds to an equivalence class of descriptions (i.e. SHGCs), we will call two descriptions *equivalent* when they describe the same solid shape, as did Marr and Nishihara in [6], hence, we refer to this problem as the *Equivalence Problem*. We will actually use the term "equivalent" in this paper when the ends are slanted differently, as long as the shapes are otherwise the same.

There are four trivial changes possible in the s and t coordinates themselves while preserving equivalence:

- the axis can be flipped end-over-end to yield a new SHGC (reversing the sense of the s coordinate)
- the sense of t can be likewise reversed and, if the cross-section $\mathbf{C}(t)$ is closed, the point at which $t = 0$ can be shifted to anywhere on the curve
- the radius function $r(s)$ can be multiplied by any constant scale factor, while the cross-section $\mathbf{C}(t)$ is divided by the same factor
- an RSHGC can have the u-v axes rotated about the origin arbitrarily (shifting the t coordinate).

These transformations are sufficiently simple that no deeper discussion is needed.

There are, however, more significant variations in the possible descriptions of a specific shape as an SHGC. We will investigate two of the principal types of variation: altering the orientation of the cross-section planes and altering the direction of the axis.

4.1 The Equivalent Right SHGC Problem

What properties of a shape make it possible to describe it as two different SHGCs, with cross-section planes at different orientations? Since this question is so general, we will limit our attention to a more restricted (but still difficult) question: For what SHGCs are there equivalent Right SHGCs? This is interesting since the RSHGC seems to be a natural "canonical" form of representation for a shape. We will ignore the effect of "beveled" ends resulting from values of α not equal to $\pi/2$.

To make this problem somewhat more tractable, we will presume that the same axis A and radius function r are to be used for the SHGC and RSHGC. (We conjecture, but have not proven, that this presumption implies no loss of generality.) The problem can then be stated this way: Given an SHGC $G_1 = (\mathbf{A}, \mathbf{C}_1, r, \alpha)$, with $\mathbf{C}_1 = (u_1,v_1)$, can some function $\mathbf{C}_2 = (u_2\, v_2)$ be found such that the RSHGC $G_2 = (\mathbf{A}, \mathbf{C}_2, r, \pi/2)$ contains the same points as G_1? This is addressed in the *Slant Theorem*:

> A non-degenerate, Oblique SHGC G_1 has an equivalent Right SHGC G_2 if and only if G_1 is Linear (figure 5).

A proof of the Slant Theorem is presented in [8].

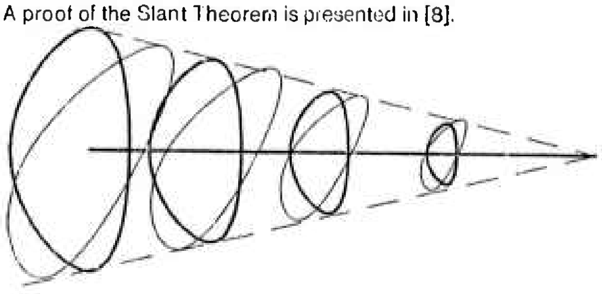


Figure 5: The Slant Theorem

So, for each LSHGC, there exists another description of the same shape which is both an LSHGC and RSHGC, containing all the same points (but without beveled ends). In this sense, the set of LSHGCs is a subset of the set of RSHGCs. Also, it doesn't matter what direction the cross-section planes are taken relative to the axis of an LSHGC: for any direction, some cross-section function $\mathbf{C}$ can be found to describe the shape as an SHGC (ignoring the possible beveling of the ends).

On the other hand, if an SHGC is not Right or Linear, then there is no Right SHGC which contains the same points.

4.2 The Alternate Axis Problem

Having addressed the issue of changing the cross-section planes, we can ask about moving the axis: For what SHGCs are there equivalent representations with different axes, using the same cross-section planes? (This is known to involve a loss of generality with respect to the question: For what SHGCs are there equivalent representations with different axes? For example, a sphere satisfies the latter condition, but not the condition we are addressing here. We conjecture that only shapes resembling certain regular polyhedra, of which the sphere is the limiting case, are excluded from our analysis herein by the restriction to use the same cross-section planes.) We will begin by restricting the problem so that the two axes intersect somewhere and so that both axes intersect the cross-section planes (i.e. the axes of the SHGCs are not parallel to the cross-section planes).

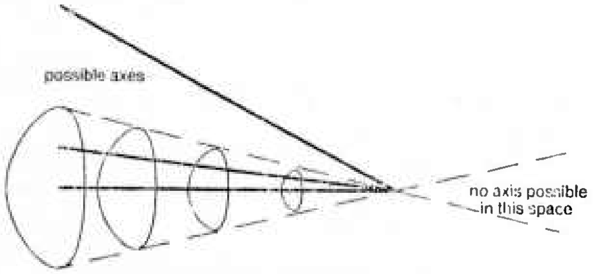


Figure 6: The Pivot Theorem

This situation is addressed by the *Pivot Theorem*:

> A non-degenerate SHGC can be described as another SHGC with a different, intersecting axis, and the same cross-section planes, if and only if it is Linear. If it is Linear, then it can be so described using any axis which passes through the apex of the shape and does not lie in the image of the shape projected through the apex (figure 6).

As in the Slant Theorem, the different representations of the shape may have different beveling of the ends. The important consequences of the Pivot Theorem are that a Linear SHGC can have (almost) any axis passing through its apex and that any non-Linear SHGC can have only one possible axis under the conditions stated above. A proof of the Pivot Theorem is presented in [8].

5. Contours of Tangency for Right SHGCs

Suppose we have an SHGC and we project it along the direction of a vector $V_E = (\Delta w_E, \Delta v_E, \Delta s_E)_{wvs}$ (a *line of sight*), as shown in figure 7. The arcs along which the surface is tangent to the line of sight as seen from direction V_E (i.e. occlusion, or parallelism to V_E) will be projected by the ends of the SHGC, or where $N \perp V_E$, i.e. $N \cdot V_E = 0$. The points on the SHGC projected onto such contours are called *contour generators* [5]. (Of course, if the SHGC is opaque, some of the contours may be hidden from view.) Contours are important because they are the usual loci of discontinuities of brightness, texture gradients, etc. in an image of a curved surface.

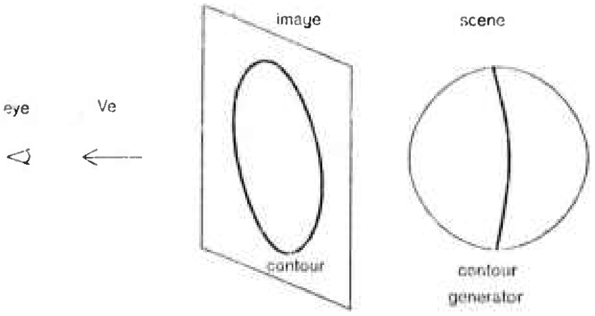


Figure 7: Contours and Contour Generators

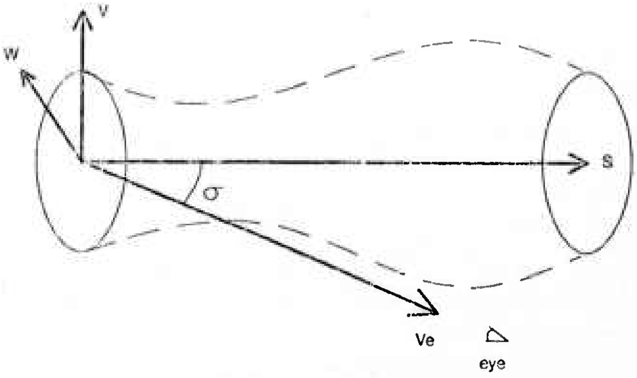


Figure 8: Viewing Direction and Angle

The discussion of imaging in this paper will be primarily limited to *orthographic projection*, in which all lines of sight are parallel. Also, unless otherwise stated, we will presume in the following discussion of projection and imaging that we are dealing only with Right SHGCs, i.e. SHGCs with cross-sections perpendicular to the axis. As shown in figure 8, this allows the simplification of rotating the $w\text{-}v$ axes as desired; in particular, we will presume that $\boldsymbol{V}_E$ is in the $\boldsymbol{W}\text{-}\boldsymbol{S}$ plane, i.e. $\Delta v_E = 0$. Without loss of generality, we can then presume that $\boldsymbol{V}_E$ is between $-\boldsymbol{W}$ and S; if the angle from $\boldsymbol{V}_E$ to $\boldsymbol{S}$ (the *viewing angle*) is σ, then $\boldsymbol{V}_E = (-\sin\sigma, 0, \cos\sigma)$. Additional simplification arises for an RSHGC since $\sin\alpha = 1$ and $\cos\alpha = 0$. For an RSHGC, using $\boldsymbol{N} \cdot \boldsymbol{V}_E = 0$, the contour generator points must satisfy

$$0 = \sin\sigma \frac{dv_C}{dt} + h(t)\cos\sigma \frac{dr}{ds} \qquad (5\text{-}1)$$

In an end view or side view ($\sigma = 0$ or $\sigma = 90^\circ$), the contour generators are always planar, but in an oblique view, the contour generators are generally not planar unless the shape is a Linear Right SHGC (see the discussion in [8]).

5.1 Images of Right SHGCs

The assignment of world (scene) coordinates is shown in figure 9. It involves aligning $\boldsymbol{V}$ vertically ($V = Y$), and placing the line of sight V_E on $\boldsymbol{Z}$ ($V_E = Z = (0,0,1)$). (Recall here that un-subscripted coordinates are given in the $x\text{-}y\text{-}z$ system.) Then $\boldsymbol{S}$ and $\boldsymbol{W}$ are in the horizontal ($\boldsymbol{X}\text{-}\boldsymbol{Z}$) plane. Note here that the origin is at $(0,0,0)_{wvs}$ rather than at the eye; under orthography, this changes no important geometric relationships.

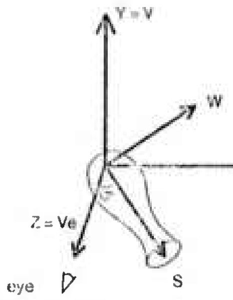


Figure 9: Object and World Coordinate Systems

The image of a point $P(s,t)$ on an RSHGC under orthography is thus

$$I(s,t) = (x,y)\,(s,t) = (u_C(t)\,r(s)\cos\sigma + s\sin\sigma,\; v_C(t)\,r(s))$$

We are presuming here that there is no scaling difference between w-v-s and x-y-z coordinates.

5.2 Contour Generators Under Perspective Projection

We can also analyze images of RSHGCs under perspective projection. The contour generator analysis itself can be accomplished by considering the eye (center of the lens) to be located at a point $P_E = (w_E, v_E, s_E)$ in the object-centered coordinate system, as in figure 10. Then, at each point $P(s,t)$ on the surface of the object, the line of sight is the vector $V_E(s,t)$ from the eye to $P(s,t)$, defined by:

$$\begin{aligned} V_E(s,t) &= P(s,t) - P_E \\ &= (u_C(t)\,r(s) - w_E,\; v_C(t)\,r(s) - v_E,\; s - s_E) \end{aligned}$$

Along a contour generator, we still have $N \perp V_E$, so $0 = N \cdot V_E = N \cdot (P(s,t) - P_E)$.

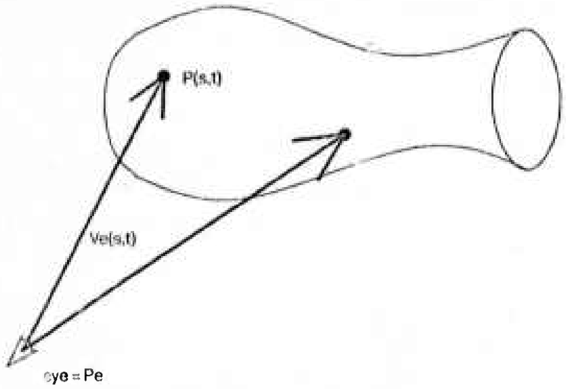


Figure 10: Imaging Under Perspective Projection

6. Contour Formation by Right Circular SHGCs

For solids of revolution, which are Right Circular SHGCs, there is considerable simplification in the orthographic projection and imaging relationships. Recall that, for a Circular SHGC, $u_C(t) = \cos 2\pi t$ and $v_C(t) = \sin 2\pi t$. Using equation (5-1), the contour generators for an RCSHGC must satisfy

$$0 = \sin\sigma\cos 2\pi t + \cos\sigma\,\frac{dr}{ds}$$

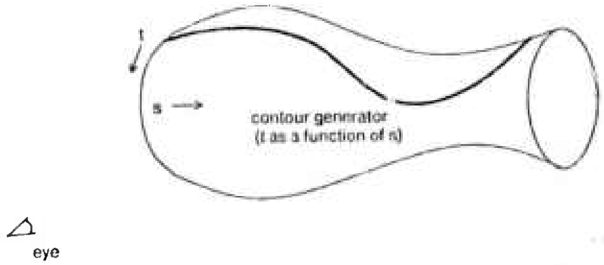


Figure 11: Contour Generator on a Right Circular SHGC

and therefore, as shown in figure 11,

$$t = \frac{1}{2\pi}\cos^{-1}\left(-\cot\sigma\,\frac{dr}{ds}\right) \qquad (6\text{-}1)$$

This equation is of fundamental importance, since it expresses t as a function of s along the contour generator. Thus, given a radius function $r(s)$ of a solid of revolution and a viewing angle σ, the above equation tells exactly how the contour generator moves towards and away from the viewer.

Now, since t is a function of s along the contour generator, the points $P(s,t)$ along the contour generator can be specified as $P_{CG}(s)$, a function of s only:

$$P_{CG}(s) = \left(\cot\sigma\, r(s)\frac{dr}{ds},\; \pm r(s)\sqrt{1 - \cot^2\sigma\,(dr/ds)^2},\; s\right)$$

The contour generator includes points such that the v-coordinate of $P_{CG}(s)$ is defined, i.e. $|dr/ds| \leq |\tan\sigma|$. The contour generator is not generally planar in an oblique view.

On an RCSHGC, the image of a point $P_{CG}(s)$ on the contour generator is

$$\begin{aligned} I_{CG}(s) &= (x_{CG}, y_{CG})(s) \\ &= \left(-r(s)\,\frac{\cos^2\sigma}{\sin\sigma}\frac{dr}{ds} + s\sin\sigma,\; r(s)\sqrt{1 - \cot^2\sigma\,(dr/ds)^2}\right) \quad (6\text{-}2) \end{aligned}$$

Further, the slope of the image contour, dy_{CG}/dx_{CG}, can be determined as a function of dr/ds using the above equation:

$$\frac{dy_{CG}}{dx_{CG}} = \left(1 \Big/ \sqrt{\sin^2\sigma - \cos^2\sigma\,(dr/ds)^2}\right)\frac{dr}{ds} \qquad (6\text{-}3)$$

6.1 Occlusions and Singularities in Image Contours

Where $|dr/ds| > |\tan\sigma|$, there will be no points on the contour generator. This causes occlusion of the contour generator from view, with resulting discontinuity in the visible contours.

Figure 12 shows an object seen from a side view ($\sigma = \pi/2$) In this view, $\tan\sigma$ is infinite and $|dr/ds| \leq \tan\sigma$ for all s. There will be a single continuous contour generator on the object, which will in fact be planar (running along the top and bottom of the object).

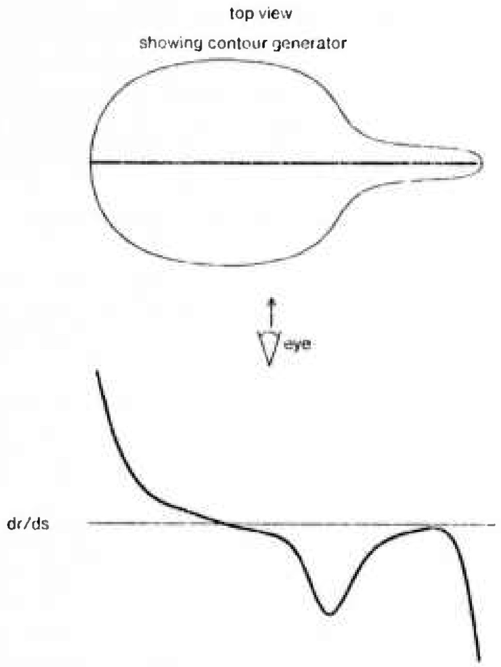


Figure 12: Contour Generator in Side View

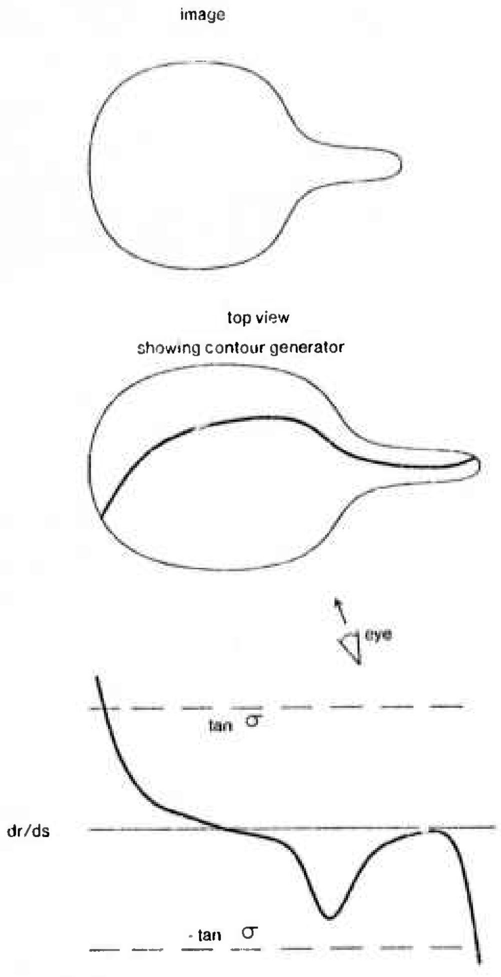


Figure 13: Contour Generator in Near-Side View

Figure 13 shows what happens as as we rotate the object slightly, decreasing σ and hence $tan\ \sigma$. As long as $|dr/ds| \leq tan\ \sigma$ everywhere, the contour generator will still be continuous. However, it will no longer be planar (unless the object is also a Linear SHGC, which we will not consider further here). From equation (6-1), we see that where dr/ds is 0, $t = 1/4$, i.e. the contour generator is on "top" of the object. (Of course, there is also an identical contour generator on the bottom.) Where $dr/ds < 0$, $t > 1/4$ and the contour generator is pushed away from us; where $dr/ds > 0$, the contour generator is pulled towards us. Note also that $|dr/ds| > |tan\ \sigma|$ at the ends of the object, hence the contour generator no longer includes the ends as the object is slightly rotated away from a side view.

Let us presume for the moment that the object is thinner at the near end, i.e. $dr/ds \leq 0$ towards that end. Eventually, as shown in figure 14, we rotate the object so much that $dr/ds \doteq -tan\ \sigma$ at some value of s, say s_m, where dr/ds is at a minimum At this value, since $dr/ds\,|_{s_m} = -tan\ \sigma$ and $d^2r/ds^2\,|_{s_m} \doteq 0$ (because s_m is a relative minimum for dr/ds), we have $dx_{CG}/ds\,|_{s_m} = 0$ and $dy_{CG}/ds\,|_{s_m} = 0$. Thus, the contour generator is tangent to the line of sight at s_m.

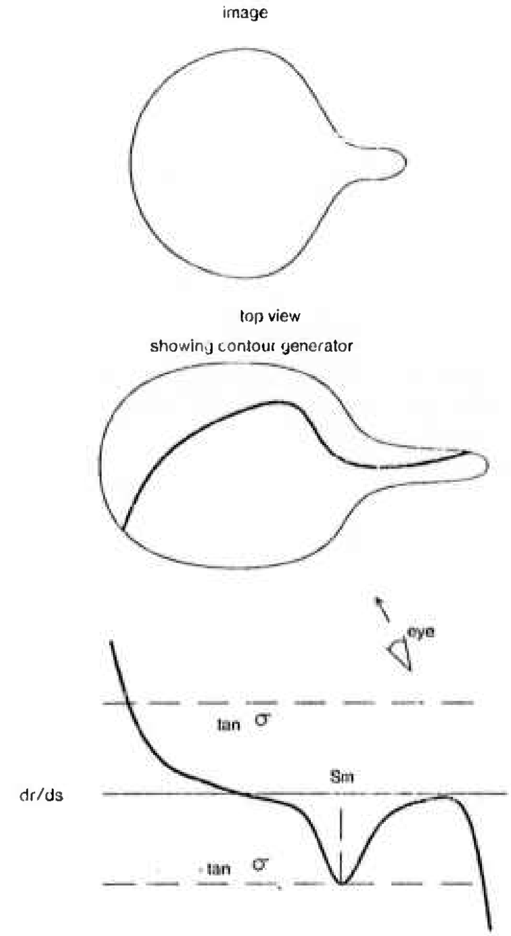


Figure 14: Contour Generator Tangent to Line of Sight

Figure 15 shows what happens when we rotate the object yet farther. There is an interval (s_a, s_b) around s_m in which $dr/ds < -tan\ \sigma$, i.e. for which no contour generator points exist. What has happened is that the former, single contour generator has been split into two separate contour generators, corresponding to values of s such that $s \geq s_b$ and $s \leq s_a$. Along the contour for $s \geq s_b$, all points will be visible in the image (i.e. none are occluded by the

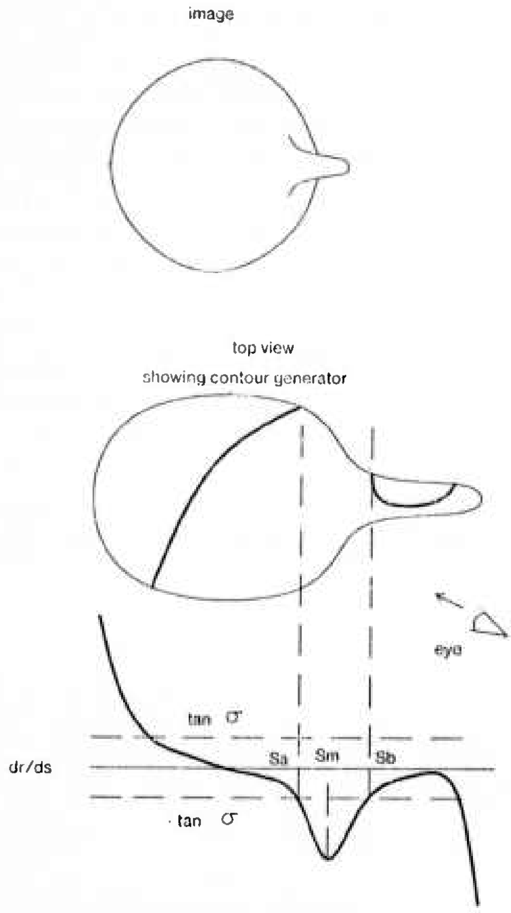


Figure 15: Contour Generator Split at Near End View

object itself in this vicinity). Meanwhile, along the contour for $s \le s_a$, tne object itself will occlude part of the contour generator, for values of s above some value s_c (where $s_c \le s_a$) (segment X in figure 16). In any event, $r(s)$ can be determined for segments A, B, and Y.

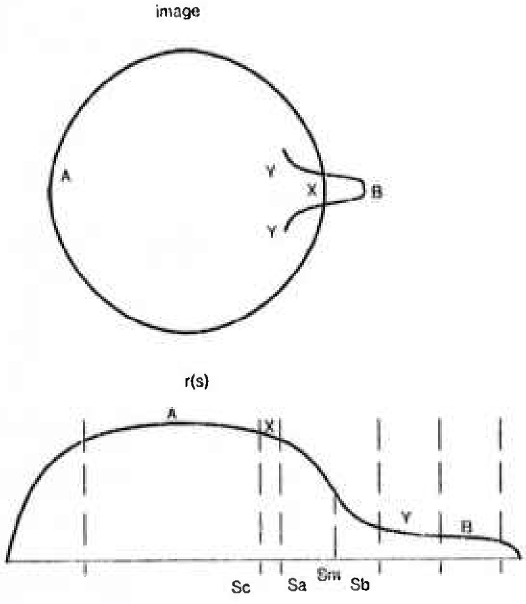


Figure 16: Contour Pieces Correspond to Disjoint Intervals of *s*

What we have seen is a single image contour splitting into two parts connected by a sort of "T" junction; the split occurred at the point at which the contour generator was tangent to the line of sight. This illustrates an important kind of "special viewpoint" for curved surfaces: If a visible curve in the scene is tangent to the line of sight, then a small variation in the viewpoint can cause a topological change in the image of the curve.

7. Contour and Silhouette Analysis for Right Circular SHGCs

In image understanding, we have the task of analyzing image contours rather than predicting them. We can use the properties of contour generators, as described above, to derive analysis techniques.

Suppose we have a line drawing consisting of visible (i.e. unoccluded) contours, each of which is the image of some contour generator on a Right Circular SHGC. By analyzing the contours, we can construct a description of the solid shape portrayed.

First, we need to determine the viewing angle σ and to align the image as in figure 17, so the images of the endpoints of the axis $A(0)$ and $A(1)$ are at (0,0) and ($\sin \sigma$, 0), respectively; this conforms to the imaging model presented previously. Then, we can analyze the contours to recover the shape; for a Right Circular SHGC, we need only determine $r(s)$, the radius function, to have a complete description of the shape.

We will begin by addressing the latter problem -- analysis of contours when the image is aligned and σ known. Then, we will examine how to determine σ and perform the alignment.

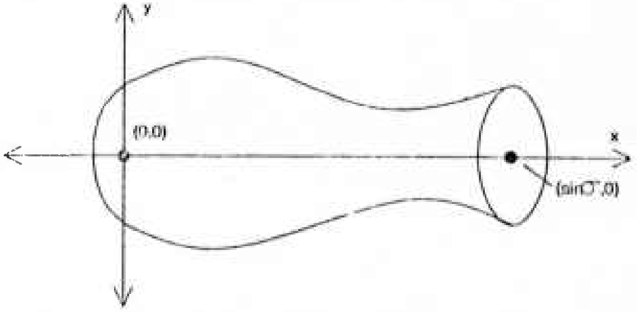


Figure 17: Aligned Image of an SHGC

7.1 Contour Analysis

Along a contour generator of a Right Circular SHGC, recall that equations (6-2) and (6-3) give $x_{CG}(s)$, $y_{CG}(s)$, and dy_{CG}/dx_{CG} as functions of s, $r(s)$, and dr/ds. These allow us to solve for s, $r(s)$, and dr/ds, the shape description, as functions of $x_{CG}(s)$, $y_{CG}(s)$, and dy_{CG}/dx_{CG}, which can be measured in the image.

$$s = \frac{x_{CG}(s) + y_{CG}(s) \cos^2 \sigma \, (dy_{CG}/dx_{CG})}{\sin \sigma}$$

$$r(s) = y_{CG}(s) \sqrt{1 + \cos^2 \sigma \, (dy_{CG}/dx_{CG})^2}$$

$$\frac{dr}{ds} = \left(\sin \sigma \, / \sqrt{1 + \cos^2 \sigma \, (dy_{CG}/dx_{CG})^2} \right) \frac{dy_{CG}}{dx_{CG}}$$

So, given any contour point ($x_{CG}(s)$, $y_{CG}(s)$), and the slope dy_{CG}/dx_{CG} of the contour at that point, we can determine s, $r(s)$, and dr/ds. By doing this for all points on all contours, we can determine as much as possible about $r(s)$ (figure 18).

As stated earlier, the above analysis presumes that σ is known and that the image is appropriately aligned. Figure 19 shows that, if σ is not known in advance, the image of the contours of a solid of revolution is ambiguous, with nne degree of freedom (σ). Part (a) of figure 19 shows a solid of revolution from a side view; (b) shows the image contours produced when the shape is viewed from a viewing angle of 45°. Part (c) shows the set of different solids of revolution which might have produced the figure if viewed from various angles. (Thick lines in (c) correspond to visible contours; thin lines are portions of $r(s)$ estimated by the method **below.** For each possible value of σ within some interval of possibilities, a plausible function $r(s)$ can be determined whose image contours match those actually seen.

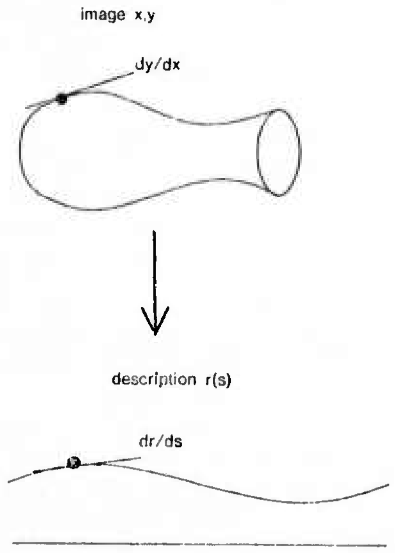


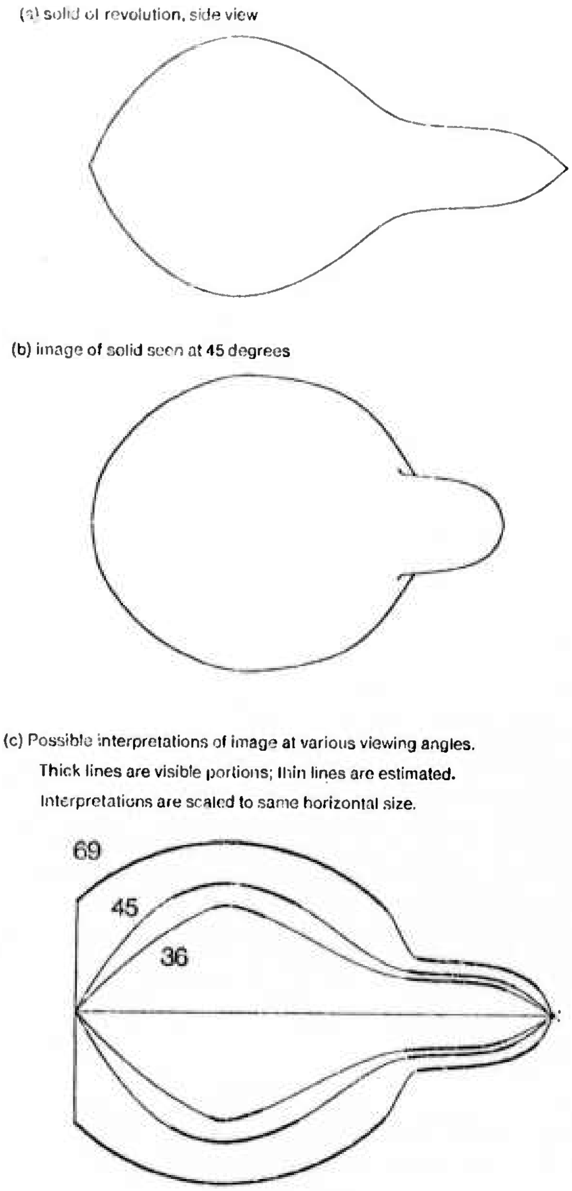
Figure 18: Contour Analysis Results in Shape Description

7.2 Occluded Contours and Silhouettes

We have seen how to analyze contours to determine values of $r(s)$; now, we will discuss how much of $r(s)$ can be reconstructed in this manner (i.e. over what range of values of s). As we already know, there is a contour generator only where $|dr/ds| \le |tan\ \sigma|$; values of s for which $|dr/ds| > |tan\ \sigma|$ therefore do not correspond to any points on a contour generator, and $r(s)$ cannot be determined for these values by examination of image contours. In addition, as described in section 6.1, the object itself may occlude portions of the contour generator from view.

For analyzing a silhouette, exactly the same methods and conditions apply, except that, using the notation of section 6.1, the contour for $s \le s_c$ will render invisible the contour for $s \ge s_b$ which lies to the left of $I_{CG}(s_c)$; thus, in figure 16, only segments A and B will be visible. Silhouettes are simply images of contours in which certain portions of some contours are not visible.

If $dr/ds > 0$, the situation is just the above viewed from the opposite direction (i.e. segments A, B, and X in figure 16 will be visible and segment Y will be occluded). In this case, when the contour generator splits, the arc for $s \le s_a$ is still occluded, but the arc for $s \ge s_b$ is a closed curve in the image rather than flaring out as above. Silhouette analysis will be identical; indeed, the silhouette of an object is identical (to within a reflection) viewed from opposite directions.



(a) solid of revolution, side view

(b) image of solid seen at 45 degrees

(c) Possible interpretations of image at various viewing angles.
Thick lines are visible portions; thin lines are estimated.
Interpretations are scaled to same horizontal size.

Figure 19: Contours of a Solid of Revolution are Ambiguous

In any of these cases, there will be an interval of s over which $r(s)$ cannot be computed, say (s_i, s_j). However, we can compute $r(s_i)$, $r(s_j)$, $dr/ds|_{s_i}$, and $dr/ds|_{s_j}$. For practical image analysis, it is possible to estimate $r(s)$ over (s_i, s_j) by fitting a function which conforms to these constraints. For example, a cubic polynomial can be fit to the data. If we let $r(s) = as^3 + bs^2 + cs + d$, we then have $dr/ds = 3as^2 + 2bs + c$. Then the following system of linear equations can easily be solved to determine the values of a, b, c, and d:

$$\begin{bmatrix} r(s_i) \\ r(s_j) \\ dr/ds\big|_{s_i} \\ dr/ds\big|_{s_j} \end{bmatrix} = \begin{bmatrix} s_i^3 & s_j^2 & s_i & 1 \\ s_j^3 & s_i^2 & s_j & 1 \\ 3s_i^2 & 2s_i & 1 & 0 \\ 3s_j^2 & 2s_j & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

7.3 Aligning the Image

The above analysis has presumed that we know the viewing angle σ and have aligned the images of the axis endpoints $A(0)$ and $A(1)$ onto (0,0) and ($sin\ \sigma$, 0), respectively. We will now address the problems of aligning the image and determining σ.

Suppose we are given an image of the contours of a Right Circular SHGC, arbitrarily scaled, rotated, and translated, and viewed from an unknown angle. We can immediately determine the image of the axis, since this will be an axis of symmetry in the image, and rotate the image so this is horizontal. By translating the image, this axis can be made to line up with the x-axis. Brooks [3] and Marr and Nishihara [6] discuss the issue of the finding the image of the axis of a generalized cylinder.

We must next determine which end of the object is nearer, so this can be placed on the right as in our imaging model. If the left end is closer, we will need to reflect the image about the y-axis, or equivalently rotate it 180° about the origin.

We must also determine σ and find the images of the axis endpoints. Determining σ is more important, since it affects both the image alignment and the shape of the reconstructed function $r(s)$. The axis endpoints, on the other hand, affect only the scale and shifting (along s) of $r(s)$.

Figure 20 shows an object whose closer end is Flat, i.e. $r(1) > 0$. The edge of the cross-section at that end will produce a contour in the image, which will be a vertically elongated ellipse. This is a very useful configuration, since we can easily determine which end of the object is closer. Then, we know that the center of the ellipse must be the image of the axis endpoint $A(1)$; further, we can compute the viewing angle σ from the eccentricity of the ellipse, using $cos\ \sigma = b/a$, where a and b are the semimajor and semiminor axis lengths, respectively.

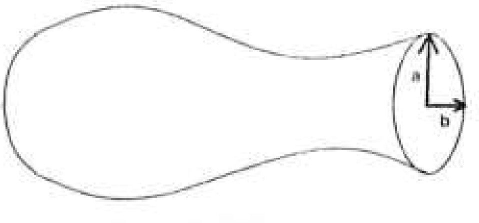


Figure 20: RCSHGC With Near End Flat

If the farther end of the object is Flat and not occluded, i.e. $r(0) > 0$ and $dr/ds\,\big|_0 < tan\ \sigma$, then we will see exactly half of an ellipse, which can be analyzed as above to determine which end of the object is farther, the image of $A(0)$, and σ. If neither end can be analyzed as above, we may be able to determine whether any "T"-shaped occlusions occur along the contour; if so, the occluding contour generator is on the nearer portion of the object. Failing this, we cannot from contours alone decide which end of the object is nearer. Fortunately, there are many other potential sources for alignment information, such as the object's length or width, or surface normals as determined by photometry or range data.

8. Conclusions

In this paper, we have explored some of the properties of Straight Homogeneous Generalized Cylinders. We began by defining Straight Homogeneous Generalized Cylinders (SHGCs) as Generalized Cylinders with straight axes and cross-sections of constant shape but varying size. With this definition as a starting point, we saw that a natural set of object-centered coordinates can be used to specify the positions of points on the surface of an SHGC and the direction of the surface normal at each point.

We looked at the Equivalence Problem, and saw that only Linear SHGCs (i.e. those with radius proportional to distance from some apex point on the axis) can be described in more than one way with different cross-sections or (under some restrictions) with different axes. This means that an an hourglass-shape, for example, can only be described as an SHGC in one way.

Next, we looked at the analysis of tangency contours of SHGCs. We began by presenting the condition which must be satisfied by the points on the contour generator of any SHGC. Right SHGCs allow an important simplification of this condition. The contour generator is planar in an end view or side view of any Right SHGC, and in any oblique view of a Right Linear SHGC. In an oblique view of a non-Linear Right SHGC, it is difficult to determine whether the contour generator is planar.

Solids of revolution (Right Circular SHGCs) are amenable to much more detailed analysis. It is possible to describe the visible contours in an image of a solid of revolution as they depend on the radius function and the viewing angle; these relationships can be inverted to determine the radius function from the image contours, when the viewing angle is known in advance. If the viewing angle is not known, the image of a solid of revolution is ambiguous with one degree of freedom. The viewing angle can be determined if the object has a visible flat end. Silhouettes can be analyzed by the same methods used for tangency contours.

It was also observed that a "special viewpoint" exists when the contour generator is tangent to the line of sight. For a solid of revolution, this occurs when the viewing angle is equal to a local maximum in the angle between the radius function and the s-axis.

9. Bibliography

1. Ballard, D. H. and Brown, C. M.. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

2. Binford, T. O. Visual perception by computer. Proc. IEEE Conf. on Systems and Control, Miami, December, 1971.

3. Brooks, R. A. "Symbolic reasoning among 3-D models and 2-D images." *Artificial Intelligence 17* (1981), 285-348. Special volume on computer vision.

4. Hilbert, D. and Cohn-Vossen, S.. *Geometry and the Imagination.* Chelsea Publishing Co., New York, 1952.

5. Marr, D. "Analysis of occluding contour." *Proc. Royal Society of London B-197* (1977), 441-475.

6. Marr, D. and Nishihara, H. K. "Representation and recognition of the spatial organization of three-dimensional shapes." *Proc. Royal Society of London B-200* (1978), 269-294.

7. Rubinstein, Z.. *A Course in Ordinary and Partial Differential Equations.* Academic Press, New York, 1969.

8. Shafer, S. A. and Kanade, T. The Theory of Straight Homogeneous Generalized Cylinders, and a Taxonomy of Generalized Cylinders. Carnegie-Mellon University Computer Science Department, January, 1983.

An Algorithm to Display Generalised Cylinders

Richard Scott

Department of Computer Science
Stanford University, Stanford Ca 94305

Abstract

This paper describes an algorithm, capable of parallel implementation, to calculate the perspective image of a Generalised Cylinder, from arbitrary viewpoint, with hidden surface removal. It applies to a wide class of cylinders. The time taken will be proportional to the total length of the contours, independent of the number of edges. The algorithm solves for one closed-loop contour-generator at a time, testing its contour (in the image plane) for intersection with visible segements of previous contours. The input are the functions describing the object, along with the position of the eye; and the output are the visible surfaces and edges.

1. Introduction

Model based vision systems predict the appearence of their models by calculating the perspective projection or "image", from different viewpoints. In the past, this has only been done for models with analytic, closed form solution to the perspective projection equations. Since Generalised Cylinders are a natural, and commonly used model, a general method for getting their image will be useful. From a 3-D Computer Graphics viewpoint, the algorithm presents an alternative to building models out of surface patches and planar surfaces and it is equally general. Most previous methods for displaying parametric surfaces have ignored the structure of the scene; for example scan line algorithms can be fooled by certain folds in the surface.

The overall idea of the algorithm is to look for those lines on the models' surface whose points are tangent to the line of sight, called "Contour-Generators". These form closed loops. If one point is known it can be efficiently propagated to form the whole line, by using a simple expression for the contour-generator tangent vector. Surfaces become occluded when they pass behind contour-generators. So by calculating where the perspective projections of the contour-generators, called "contours", cross each other, the visible outlines of the model are found. The visible surfaces are produced by expanding the regions attached to the front side of each contour-generator until their image crosses a contour. The method is a generalisation of one used in [3], for volumes bounded by planar faces.

The image of a scene consisting of several larger objects, made up of intersecting models, is obtained one model at a time. Negative volumes or holes, in objects can also be displayed. The models themselves can have smoothly changing surfaces and edges.

The rest of the paper will be split into:

2. Generalised Cylinder definition

3. Outline of the algorithm.

4. Implementation of the definition in 2., and discusion of those Generalised Cylinders accepted by the algorithm, (almost all of them).

5. Results, equations, formulas, referred to in the algorithm outline 3., and Generalised Cylinder derivative functions.

6. Comparison to other Algorithms.

7. Current state of the implementation.

8. Extensions and summary.

2. Generalised Cylinder Definition

Generalised Cylinders were first introduced by Binford [4]. When a planar shape sweeps through space, the volume it passes through is a Generalised Cylinder (GC). Recently Shafer [2], has made a precise definition, which I give here.

(i) There is a space curve, called the axis of the shape.

(ii) At each point in the axis, at some fixed angle to the axis, there is a cross-section plane defined.

(iii) On each such cross-section plane, there is a planar curve which constitutes the cross-section of the object on that plane.

(iv) There is a deformation rule which specifies the transformation of the cross-section as the cross-section plane is swept along the axis. This rule always imposes (at least) the constraint that the cross-section changes smoothly.

(v) I add the further condition that Every point in the object must lie on exactly one cross-section plane.

This is the class of shapes that the algorithm deals with, along with the restriction that the cross-section curve is closed.

3. Outline of Algorithm

The algorithm can be divided into two parts. First (A), solution for the visible parts of the contour-generators, and secondly (B), region growing to get visible surfaces. The first part is the principal one. It has two subparts, which are repeated, and together find one contour-generator. Each contour-generator is a closed loop, intersecting no others, which divides the surface into forward (visible if unoccluded), and backward facing (invisible) areas. The square root of the size of each visible area, is a measure of the length scale over which things are happening in that region of the GC.

The first subpart (A1), steps over the GC with step length proportional to the square root of the area of the region it is contained by, until either the whole surface has been covered, in which case the algorithm stops; or a step containing a new contour-generator is found. In this case, the step is then bisected down to an exact solution. A test to see whether each step jumps a new contour-generator can be made. For whenever the direction (forward or back), that a surface point is facing, differs from the direction predicted by the regions of the existing contour-generators, then there must be an undiscovered contour-generator passing nearby. This means that if one stepping point has the same predicted, and actual surface direction, and the next does not, then a new contour generator passes through the intervening step. This interval is reduced, using bisection, with the condition that one end of the interval must have the same predicted, and real surface directions, while the other end must not.

The solution is handed over to the second subpart (A2), which propagates it around the whole contour-generator, back to its start, making a list of the solutions as it goes, and noting the ones where the contour-generator becomes visible or occluded. It works by stepping along the contour-generator tangent (2)(see section 5.), to get a guess for the next solution point, which is Newton-Raphson iterated to a sufficient accuracy (6)(section 5.). If the Newton-Raphson does not converge, several points around a small circle are tested to find an interval to bisect down to the next solution. The step length is taken proportional to curvature of the contour, to get uniform interpolation accuracy between the known contour points (3),(4). Each step is projected to the image, and checked for intersection with these previously projected steps, which have not been shown to be hidden. When an intersection is found, the exact positions of the occluding and occluded contour-generator points are calculated(8). Finally the whole contour is checked against possible surrounding contours.

To convert to a form implementable in parallel; step A1 is done independently, at different points on the GC and then A2 is used to form contour- generator segments, which can be simply joined up into the complete contour lists.

Either way, each list of contour points is now followed down, keeping count of the marked occluded points, to produce lists of just the visible ones.

4. Implementation of Definitions

The definition, from section 2, can be implemented as a bivariate surface, with s parametrising the axis and t parametrising the cross-section curve, both roughly proportional to arc length, $0 \leq s, t \leq 1$. If N denotes the outward normal, (t , s , N) form a right handed set. The quantities that have to be defined are:

E   the position vector of the eye.

SP(s)   the axis (or spine) (can use intrinsic coordinates) $0 \leq s \leq 1$.

α the angle between the y-axis of the cross-section plane and the spine.

C(t,s)   the closed cross section curve. Given s, t parametrises the curve so that $C(0,s) = C(1,s)$, with t roughly proportional to arc length.

Functions for the deformation rule:

tx(s)   a twisting angle, giving rotation of the cross-section in its own plane.

Derivable from these are:

P(t,s)   a point on the GC surface.

$F(t,s) = P - E$ the line of sight vector,

N(t,s)   the normal to the surface.

$N.F = 0$ is the equation to be solved. (See the results and formuals section) where . means scalar product.

The algorithm will work for arbitrary cross-section C(t,s). But any C(t,s) can be decomposed, (to desired accuracy by taking enough terms), into separable form, by

$$C(t,s) = r_1(s)^*C_1(t) + r_2(s)^*C_2(t) + r_3(s)^*C_3(t) + \ldots$$

where the r_i are called "radius" functions.

To see this take, for example, $C_i(t) = C(t, \frac{i}{n})$ for $0 \leq i \leq n$ and take $r_i(s)$ to interpolate smoothly between the cross-section C_i functions. Then as $n \to \infty$, the *expression on rhs* $\to C(t,s)$. This decomposition also has useful properties when a small number of terms are taken. eg. with two terms, a cross-section shape can be stretched in two independent directions; or ruled surfaces can be formed; or a square can be smoothly deformed into a circle (this is the shape of many coffee jars).

For example the following functions merge a square into a circle, along a parabolic axis.

$$C_1 = (cos2\pi t, sin2\pi t)$$
$$C_2 = (1 - 4t, 4t) \quad 0 \leq t \leq \tfrac{1}{4}$$
$$C_2 = (1 - 4t, 2 - 4t) \quad \tfrac{1}{4} \leq t \leq \tfrac{1}{2}$$
$$C_2 = (-3 + 4t, 2 - 4t) \quad \tfrac{1}{2} \leq t \leq \tfrac{3}{4}$$
$$C_2 = (-3 + 4t, -4 + 4t) \quad \tfrac{3}{4} \leq t \leq 1$$

$$r_1 = s$$

$$r_2 = 1 - s$$

$$SP = (20s, 10s^2, 0)$$

Generalised Cylinders can be built up into more complicated models. They can describe the shape of additions and also of holes. "Negative volume", that is, a hole in a solid object, can be displayed in the same way as everything else. This is done by distinguishing two types of contour-generator points, depending on how the line of sight makes a tangent there. The possibilities are, that the tangent lies outside the surface of the model, "outer" contour-generator points, or that it lies inside, "inner" contour generator point. At these outer points, the adjacent forward facing surface is closer to the eye than the back face; the opposite is true for inner points. A contour-generator can often change from consisting of outer to inner points, especially in a complex scene. To give a generalised cylinder negative volume, and display it as a hole, the outer and inner contours need to be swapped, and the outward pointing vectors at the cutoff ends are reversed.

5. Results, Formulas

This section contains some of the theory that makes the algorithm work and also the results and derivation of equations referred to in the algorithm outline.

1. Formula for P(t,s), a point in the GC surface.

$$P = SP + [R]^*[rotX, 90 - \alpha]^*[rotZ, tx]^*C(t, s) \quad (1)$$

Where SP(s) tracks out the spine as s changes, $0 \leq s \leq 1$; [R(s)] represents the rotation part of the transformation matrix into the untwisted coordinate frame on the spine; and tx is the twisting.

Let $A = \frac{dSP}{ds}$, and (a b c) $= \frac{A}{|A|}$ where $|A| \neq 0$. Then (a b c) is a unit vector tangent to the spine.

[R] is untwisted means that it represents a rotation about an axis perpendicular to the z-axis. It rotates the z-axis into (a b c). It works out to be,

$$[R] = \begin{pmatrix} 1 - \frac{a^2}{1+c} & \frac{-ab}{1+c} & a \\ \frac{-ab}{1+c} & 1 - \frac{b^2}{1+c} & b \\ -a & -b & c \end{pmatrix}$$

2. Formula for the outward surface normal, N(t,s)

$N(t, s) = \frac{\partial P}{\partial t} X \frac{\partial P}{\partial s}$ where X means vector product.

3. Formula for the tangent to a contour-generator, in parameter space.

For contour-generator s(t), $N(t, s).F(t, s) = 0$, where the dot . represents the scalar product and $F = P - E$ is the line of sight. Differentiating this with respect to t, gives a vector tangent to the contour-generator.

$$U \equiv (\frac{\partial(N.F)}{\partial s}, \frac{-\partial(N.F)}{\partial t}) \quad (2)$$

$$V \equiv (V_t, V_s) \equiv \frac{U}{|U|}$$

Given one point, P(t,s), on a contour-generator, V the unit tangent vector is used to get a good estimate for another nearby.

$$P(t + \lambda^* V_t, s + \lambda^* V_s) \quad (3)$$

Where λ is the step length.

4(i) The vector V always points the same way. Imagine looking down at the contour-generator from outside the cylinder, then the vector has the forward face on its right, and the back face on its left, when (t, s, N) form a right handed set.

Proof: either by drawing four diagrams (see diagrams); or, since U changes continuously, parallel to the contour- generator, change of direction means that $U = 0$, and that there is another contour-generator at that point. So a contour- generator branch can be chosen to keep U pointing the same way.

4(ii) Adjacent contour-generators point in opposite directions. This is because, if the intervening surface faces forward, say, then it must be on the left hand side of both contour-generators, making them point oppositely.

4(iii) Contour-generators cannot cross. Since crossing means that U = 0, (see diag), take the continuation of the contour-generator to be either L or R, whichever makes the smallest angle between steps.

5. Step length, λ, along V is taken proportional to the curvature in the image plane. This means that interpolation in the image plane, joining up the calculated

points (by cubic spline, or whatever), will be uniformly accurate.

Step-length proportional to curvature, amounts to choosing step-length so that there is a constant angle between steps. A simple way to achieve this is,

step-length := (previous-step-length) * $\frac{A}{C}$. (4)
where A = desired angle between steps
B = actual angle between steps
C = angle between previous step and U.

C is used as an approximation for B.

6. 2-D Newton-Raphson is used to solve $N.F = 0$, starting with the guess produced by equations (3) and (4) above.

$$(t,s)' = (t,s) - \frac{Uperp(t,s)^{*} N.F(t,s)}{U.U} \qquad (5)$$

where $Uperp = (\frac{\partial(N.F)}{\partial t}, \frac{\partial(N.F)}{\partial s})$

7. The necessary derivatives for U in (2), are

$$\frac{\partial N.F}{\partial t} = F.\frac{\partial N}{\partial t} + N.\frac{\partial P}{\partial t}$$
$$\frac{\partial N.F}{\partial s} = F.\frac{\partial N}{\partial s} + N.\frac{\partial P}{\partial s}$$

where,

$$\frac{\partial N}{\partial t} = \frac{\partial^2 P}{\partial t^2} X \frac{\partial P}{\partial s} + \frac{\partial P}{\partial t} X \frac{\partial^2 P}{\partial s \partial t}$$
$$\frac{\partial N}{\partial s} = \frac{\partial^2 P}{\partial s \partial t} X \frac{\partial P}{\partial s} + \frac{\partial P}{\partial t} X \frac{\partial^2 P}{\partial s^2}$$

This reduces the problem to finding the first and second derivatives of P(t,s). Equation (1) can be rewritten as, $P(t,s) = SP(s) + [R(s)]^{*}G(t,s)$ where G(t,s) is the column vector $[rotX, 90 - \alpha]^{*}[rotZ, tx(s)]^{*}C(t,s)$

The remaining derivatives are not given here, but can quickly be worked out in this form.

8. When two contour-generator segments on the GC intersect in the image plane, the parameters of the occluded and occluding points are interpolated for. Interpolation is repeated until the images of the two points on the GC are sufficiently close.

9. Contour-generators form into a number of closed and non-intersecting (possibley touching) loops.

This happens because,

in the coordinate system of $t \rightarrow X$, $s \rightarrow Y$, $N.F \rightarrow z$, consider the surface $Z = N.F(t,s)$. It is continuous and periodic, and intersects the plane $N.F = 0$ in closed loops.

6. Comparison of Algorithms

'The Theory of Straight Generalised Cylinders' [1], contains analysis to get the image contours in closed form for generalised cylinders, with straight spines and simple radius functions. The problem is also solved in closed form for simple generalised cylinders, in Brooks' ACRONYM. So these two obtain contours for a restricted class of cylinders.

Hornung [2] uses an iterative method to solve for contour-generators, with hidden line removal, for volumes bounded by planar faces. He presents efficiency arguments which carry over to the continuous case.

Most Graphics work on parametric surfaces uses scan line methods, which can fail on certain overhanging folds in the surface. In contrast this algorithm, by explicitly working out the contours of the folds, should not be fooled.

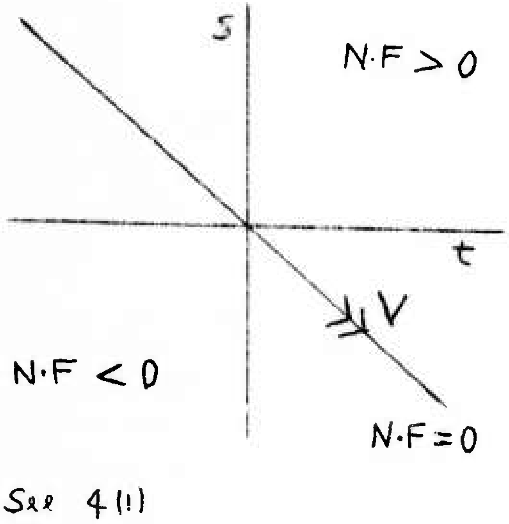
7. State of the Implementation

At present, I have implemented in maclisp, the scheme to find contour-generators, without detecting for intersection in the image plane. A backeting system, in the form of quad-trees in the image plane, and also the (t,s) plane, will provide an efficient method for detection of intersections. When each contour-generator has been completed, region growing in the (t,s) plane can fill the forward facing surface. In the image plane the inside of each contour is also filled, and this is used for detecting occlusions (see section 5, part 8.), between the contour presently being solved and previous ones. Occlusions of a contour-generator by winding back on itself, are discovered when its interior in the image plane is filled and found to have more than one region.

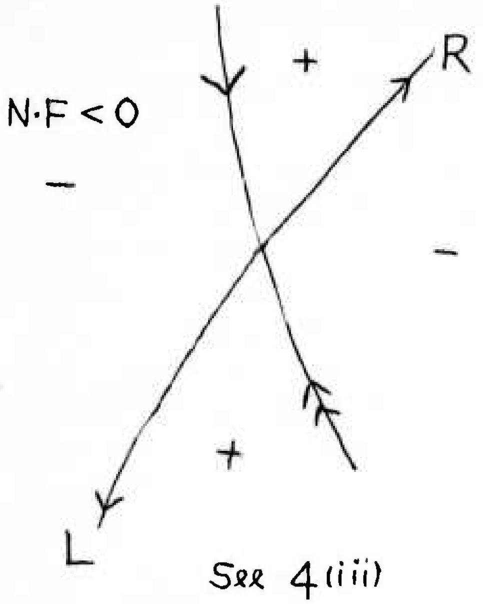
8. Future Plans, Summary

An algorithm for displaying generalised cylinders has been presented. When completely implemented, it should be robust and fast, making it useful for model based vision systems and 3-D graphics.

To extend the use of this algorithm to complex scenes, a new algorithm, to calculate the intersection line of overlapping generalised cylinders will be developed. A further extension, is to investigate symbolic and topological simplifications, which might predict the shape of the image with less numerical accuracy, but greater understanding of its structure.



See 4(i)



See 4(iii)

9. Acknowledgements

Thanks to Dave Marimont for looking over previous drafts of this paper.

This research was supported by ARPA contract number N00039-82-c-0250.

References

[Shafer 1982]
Shafer, S. A., Kanade, T., *The Theory of Straight Generalised Cylinders* .

[Shafer 1982b]
Shafer, S. A., *A Taxonomy of Generalised Cylinders* .

[Hornung 1982]
Hornung, C., *An Approach to a Calculation-Minimised Hidden Line Algorithm*, Comput.&Graphics Vol.6, No.3 '82.

[Binford 1971]
Binford, T. O., *Visual Perception by Computer*, Proc. IEEE Conf. on Systems and Control. Miami, December '71.

PERCEPTUAL ORGANIZATION AND CURVE PARTITIONING*

By: M.A. Fischler and R.C. Bolles

SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

ABSTRACT

In this paper we offer a critical evaluation of the partitioning (perceptual organization) problem, noting the extent to which it has distinct formulations and parameterizations. We show that most partitioning techniques can be characterized as variations of four distinct paradigms, and argue that any effective technique must satisfy two general principles. We give concrete substance to our general discussion by introducing new partitioning techniques for planar geometric curves, and present experimental results demonstrating their effectiveness.

I INTRODUCTION

A basic attribute of the human visual system is its ability to group elements of a perceived scene or visual field into meaningful or coherent clusters; in addition to clustering or partitioning, the visual system generally imparts structure and often a semantic interpretation to the data. In spite of the apparent existence proof provided by human vision, the general problem of scene partitioning remains unsolved for computer vision. Furthermore, there is even some question as to whether this problem is meaningful (or a solution verifiable) in its most general form.

Part of the difficulty resides in the fact that it is not clear to what extent semantic knowledge (e.g., recognizing the appearance of a straight line or some letter of the English alphabet), as opposed to generic criteria (e.g., grouping scene elements on the basis of geometric proximity), is employed in examples of human performance. It would not be unreasonable to assume that a typical human has on the order of tens of thousands of iconic primitives in his visual vocabulary; a normal adult´s linguistic vocabulary might consist of from 10,000 to 40,000 root words, and iconic memory is believed to be at least as effective as its linguistic counterpart. Since, at present, we cannot hope to duplicate human competence in semantic interpretation, it would be desirable to find a task domain in which the influence of semantic knowledge is limited. In such a domain it might be possible to discover the generic criteria employed by the human visual system and to duplicate human performance. One of the main goals of the research effort described in this paper is to find a set of generic rules and models that will permit a machine to duplicate human performance in partitioning planar curves.

II THE PARTITIONING PROBLEM: ISSUES AND CONSIDERATIONS

Even if we are given a problem domain in which explicit semantic cues are missing, to what extent is partitioning dependent on the purpose, vocabulary, data representation, and past experience of the "partitioning instrument," as opposed to being a search for context independent "intrinsic structure" in the data? We argue that rather than having a unique formulation, the partitioning problem must be paramaterized along a number of basic dimensions. In the remainder of this section we enumerate some of these dimensions and discuss their relevance.

A. Intent (Purpose) of the Partitioning Task

In the experiment described in Figure 1, human subjects were presented with the task of partitioning a set of two-dimensional curves with respect to three different objectives: (1) choose a set of contour points that best mark those locations at which curve segments produced by different processes were "glued" together; (2) choose a set of contour points that best allow one to reconstruct the complete curve; (3) choose a set of contour points that would best allow one to distinguish the given curve from others. Each person was given only one of the three task statements. Even though the point selections within a task varied from subject to subject, there was significant overlap and the variations were easily explained in terms of recognized strategies invoked to satisfy the given constraints; however, the points selected in the three tasks were significantly different. Thus, even in the case of data with almost no semantic content, the partitioning problem is NOT a generic task independent of purpose.

----

* The research reported herein was supported by the Defense Advanced Research Projects Agency under Contract No. MDA 903-83-C-0027 and by the National Science Foundation under Contract No ECS-7917028.

B. Partitioning Viewed as an Explanation of Curve Construction

With respect to "process partitioning" (partitioning the curve into segments produced by different processes), a partition can be viewed as an explanation of how the curve was constructed. Explanations have the following attributes which, when assigned different "values," lead to different explanations and thus different partitions:

* Vocabulary (primitives and relations) -- what properties of our data should be represented, and how should these properties be computed? That is, we must select those aspects of the problem domain we consider relevant to our partition decisions (e.g., geometric shape, gray scale, line width, semantic content), and enable their computation by providing models for the corresponding structures (e.g., straight-line segment, circular arc, wiggly segment). We must also allow for the appropriate "viewing" conditions; e.g., symmetry, repeated structure, parallel lines, are global concepts that imply that the curve has finite extent and can be viewed as a "whole," as opposed to only permitting computations that are based on some limited interval or neighborhood of (or along) the curve.

* Definition of Noise -- in a generic sense, any data set that does not have a "simple (concise)" description is noise. Thus, noise is relative to both the selected descriptive language and an arbitrary level of complexity. The particular choices for vocabulary and the acceptable complexity level determine whether a point is selected as a partition point or considered to be a noise element.

* Believability -- depending on the competence (completeness) of our vocabulary to describe any curve that may be encountered, the selected metric for judging similarity, and the arbitrary threshold we have chosen for believing that a vocabulary term corresponds to some segment of a given curve, partition points will appear, disappear, or shift.

C. Representation

The form in which the data is presented (i.e., the input representation), as well as the type of data, are critical aspects of the problem definition, and will have a major impact on the decisions made by different approaches to the partitioning task. Some of the key variables are:

* Analog (pictorial) vs digital (quantized) vs analytic description of the curves

* Single vs multiple "views" (e.g., single vs. multiple quantizations of a given segment)

* Input resolution vs. length of smallest segment of interest

* Simply-connected (continuous) curves vs self-intersecting curves or curves with "gaps"

* For complex situations, is connectivity provided, or must it be established

* If a curve possesses attributes (e.g., gray scale, width) other than "shape" that are to serve as partitioning criteria, how are they obtained -- by measurement on an actual "image," or as symbolic tags provided as part of the given data set?

D. Evaluation

How do we determine if a given technique or approach to the partitioning problem is successful? How can we compare different techniques? We have already observed that, to the extent that partitioning is a "well-defined" problem at all, it has a large number of alternative formulations and parameterizations. Thus, a technique that is dominant under one set of conditions may be inferior under a different parameterization. Never the less, any evaluation procedure must be based on the following considerations:

* Is there a known "correct" answer (e.g., because of the way the curves were constructed)?

* Is the problem formulated in such a way that there is a "provably" correct answer?

* How good is the agreement of the partitioned data with the descriptive vocabulary (models) in which the "explanation" is posed?

* How good is the agreement with (generic or "expert") subjective human judgment?

* What is the trade-off between "false-alarms" and "misses" in the placement of partition points. To the extent that it is not possible to ensure a perfect answer (in the placement of the partition points), there is no way to avoid such a trade-off. Even if the the relative weighting between these two types of errors is not made explicit, it is inherent in any decision procedure -- including the use of subjective human judgment.

In spite of all of the previous discussion in this section, it might still be argued that if we take the union of all partition points obtained for all reasonable definitions and parameterizations of the partition problem, we would still end up with a "small" set of partition points for any given curve, and further, there may be a generic procedure for obtaining this covering set. While a full discussion of this possibility is is not feasible here, we can construct a counterexample to the unqualified conjecture based on selecting a very high ratio of the cost of a miss to a false-alarm in selecting the partition points. A (weak) refutation can also be based on the observation that if a generic covering set of partition points exists, then there should be a relatively consistent way of ordering all the points on a given curve as to their being acceptable partition points; the experiment presented in Figure 1 indicates that, in general, such a consistent ordering does not exist.

III PARADIGMS FOR CURVE PARTITIONING

Almost all algorithms employed for curve partitioning appear to be special cases (instantiations) of one or more of the following paradigms:

- * Local Detection of Distinguished Points: a partition point is inserted at locations along the curve at which one or more of the descriptive attributes (e.g., curvature, distance from a coordinate axis or centroid) is determined to have a discontinuity, an extreme value (maxima or minima), or a zero value separating intervals of positive and negative values.
- * Best Global Description: a set of partition points is inserted at those locations along a curve that allow the "best" description of the associated segments in terms of some a priori set of models (e.g., the set of models might consist of all first and second degree polynomials, with only one model permitted to explain the data between two adjacent partition points; the quality of the description might be measured by the mean square deviation of the data points from the fitting polynomials).
- * Confirming Evidence: given a number of "independent" procedures (or possibly different parameterizations of a given procedure) for locating potential partition points, we retain only those partition points that are common to some subset of the different procedures or their parameterizations.
- * Recursive Simplification: the input data is subjected to repeated applications of some transformation that monotonically reduces some measurable aspect of the data to one of a finite number of terminal states (e.g., differentiation, smoothing, projection, thresholding). The hierarchy of data sets thus produced is then processed with an algorithm derived from the previous three paradigms.

IV PRINCIPLES OF EFFECTIVE (ROBUST) MODEL-BASED INTERPRETATION

What underlies our choice of partitioning criteria? We assert that any competent partitioning technique, regardless of which of the above paradigms is employed, will incorporate the following principles.

A. Stability

The "principle of stability," is the assertion that any valid perceptual decision should be stable under at least small perturbations of both the imaging conditions and the decision algorithm parameters. This generalization of the assumption of "general position" also subsumes the assertion (often presented as an assumption) that most of a scene must be describable in terms of continuous variables if meaningful interpretation is to be possible.

It is interesting to observe that many of the constructs in mathematics (e.g., the derivative) are based on the concepts of convergence and limit, also subsumed under the stability principle. Attempts to measure the digital counterparts of the mathematical concepts have traditionally employed window type "operators" that are not based on a limiting process; it should come as no surprise that such attempts have not been very effective.

In practice, if we perturb the various imaging and decision parameters, we observe relatively stable decision regions separated by obviously unstable intervals (e.g., the two distinct percepts produced by a Necker cube). The stable regions represent alternative hypotheses that generally cannot be resolved without recourse to either additional and more restrictive assumptions, or semantic (domain-specific) knowledge.

B. Complete, Concise, and Complexity Limited Explanation

The decision-making process in image interpretation, i.e. matching image derived data to a priori models, not only must be stable, but must also explain all the structure observable in the data. Equally important, the explanation must satisfy specific criteria for believability and complexity. Believability is largely a matter of offering the simplest possible description of the data and, in addition, explaining any deviation of the data from the models (vocabulary) used in the description. Even the simplest description, however, must also be of limited complexity; otherwise or it will not be understandable and thus not believable.

By making the foregoing principles explicit, we can directly invoke them (as demonstrated in the following section) to formulate effective algorithms for perceptual organization.

V INSTANTIATION OF THE THEORY: SPECIFIC TECHNIQUES FOR CURVE PARTITIONING

In this section we offer two effective new algorithms for curve partitioning (program listings available from the authors). In each case, we first describe the the algorithm, and later indicate how it was motivated and constrained by the principles just presented. In both algorithms, the key ideas are: (1) to view each point, or segment of a curve, from as many perspectives as possible, retaining only those partition points receiving the highest level of multiple confirmation; and (2) inhibiting the further selection of partition points when the density of points already selected exceeds a preselected or computed limit.

A. Curve Partitioning Based on Detecting Local Discontinuity

In this sub-section we present a new approach to the problem of finding points of discontinuity ("critical points") on a curve. Our criterion for success is whether we can match the performance of human subjects given the same task (e.g., see Figure 1). The importance of this problem from the standpoint of the psychology of human vision dates back to the work of Attneave [1954]. However, it has long been recognized as a very difficult problem, and no satisfactory computer algorithm currently exists for this purpose. An excellent discussion of the problem may be found in in Davis [1977]; other pertinent references include Rosenfeld [1975], Freeman [1977], Kruse [1978], and Pavlidis [1980]. Results and observations akin and complementary to those presented here can be found in Hoffman [1982] and in Witkin [1983].

Most approaches equate the search for critical points with looking for points of high curvature. Although this intuition seems to be correct, it is incomplete as stated (i.e., it does not explicitly take into account "explanation" complexity); further, the methods proposed for measuring curvature are often inadequate in their selection of stability criteria. In Figure 2 we show some results of measuring curvature using discrete approximations to the mathematical definition.

We have developed an algorithm for locating critical points that invokes a model related to, but distinct from, the mathematical concept of curvature. The algorithm labels each point on a curve as belonging to one of three categories: (a) a point in a smooth interval, (b) a critical point, or (c) a point in a noisy interval. To make this choice, the algorithm analyzes the deviations of the curve from a chord or "stick" that is iteratively advanced along the curve (this will be done for a variety of lengths, which is analogous to analyzing the curve at different resolutions). If the curve stays close to the chord, points in the interval spanned by the chord will be labeled as belonging to a smooth section. If the curve makes a single excursion away from the chord, the point in the interval that is farthest from the chord will be labeled a critical point (actually, for each placement of the chord, an accumulator associated with the farthest point will be incremented by the distance between the point and the chord). If the curve makes two or more excursions, points in the interval will be labeled as noise points.

We should note here that "noisy" intervals at low resolution (large chord length) will have many critical points at higher resolution (small chord length). Figure 3 shows examples of curve segments and their classifications. The distance from a chord that defines a significant excursion (i.e., the width of the boxes in Figure 3) is a function of the expected noise along the curve and the length of the chord.

At each resolution (i.e., stick size), the algorithm orders the critical points according to the values in their accumulators and selects the best ones first. To avoid setting an arbitrary "goodness" threshold for distinguishing critical from ordinary points, we use a complexity criterion. To halt the selection process, we stop when the points being suggested are too close to those selected previously at the given resolution. In our experiments we define "too close" as being within a quarter of the stick length used to suggest the point.

After the critical points have been selected at the coarsest resolution, the algorithm is applied at higher resolutions to locate additional critical points that are outside the regions dominated by previously selected points. Figure 4a shows the critical points determined at the coarest level (stick length of 100 pixels; approximately 1/10 of the length of the curve). Figure 4b shows all the critical points labeled with the stick lengths used to determine them. (We note that this critical point detection procedure does not locate inflection points or smooth transitions between segments, such as the transition from an arc of a circle to a line tangent to the circle.)

The above algorithm appears to be very effective, especially for finding obvious partition points and in not making "ugly" mistakes (i.e., choosing partition points at locations that none of our human subjects would pick). Its ability to find good partition points is based on evaluating each point on the curve from multiple viewpoints (placements of the stick) -- a direct application of the principle of stability. Requiring that the partition points remain stable under changes in resolution (i.e., small changes in stick length) did not appear to be effective and was not employed; in fact, stick length was altered by a significant amount in each iteration, and partition points found at these different scales of resolution were not expected to support each other, but were assumed to be due to distinct phenomena.

The avoidance of ugly mistakes was due to our method of limiting the number of partition points that could be selected at any level of resolution, or in any neighborhood of a selected point (i.e., limiting the explanation complexity). One concept we invoked here, related to that of complete explanation, was that the detection procedure could not be trusted to provide an adequate explanation when more than a single critical point was in its field of view, and in such a situation, any decision was deferred to later iterations at higher levels of resolution (i.e., shorter stick lengths).

Finally, in accord with our previous discussion, the algorithm has two free parameters that provide control over its definition of noise (i.e., variations too small or too close together to be of interest), and its willingness to miss a good partition point so as to be sure it does not select a bad one.

B. Curve Partitioning Based and Detecting Process Homogenity

To match human performance in partitioning a curve, by recognizing those locations at which one generating process terminates and another begins, is orders of magnitude more difficult than partitioning based on local discontinuity analysis. As noted earlier, a critical aspect of such performance is the size and effectiveness of the vocabulary (of a priori models) employed. Explicitly providing a general purpose vocabulary to the machine would entail an unreasonably large amount of work -- we hypothesize that the only effective way of allowing a machine to acquire such knowledge is to provide it with a learning capability.

For our purposes in this investigation, we chose a problem in which the relevant vocabulary was extremely limited: the curves to be partitioned are composed exclusively of straight lines and arcs of circles. (Two specific applications we were interested in here were the decomposition of silhouettes of industrial parts, and the decomposition of the line scans returned by a "structured light" ranging device viewing scenes containing various diameter cylinders and planar faced objects lying on a flat surface.) Our goal here was to develop a procedure for locating critical points along a curve in such a way that the segments between the critical points would be satisfactorily modeled by either a straight-line segment or a circular arc. Relevant work addressing this problem has been done by Montanari [1970], Ramer [1972], Pavlidis [1974], Liao [1981], and Lowe [1982].

Our approach is to analyze several "views" of a curve, construct a list of possible critical points, and then select the optimum points between which models from our vocabulary can be fitted. For our experiments we quantized an analytic curve at several positions and orientations (with respect to a pixel grid), then attempted to recover the original model.

For each view (quantization) of the curve we locate occurrences of lines and arcs, marking their ends as prospective partition points. This is accomplished by randomly selecting small seed segments from the curve, fitting to them a line or arc, examining the fit, and then extending as far as possible those models that exhibit a good fit. After a large number of seeds have been explored in the different views of the curve, the histogram (frequency count as a function of path length) of beginnings and endings is usea to suggest critical points (in order of their frequency of occurrence). Each new critical point, considered for inclusion in the explanation of how the curve is constructed, introduces two new segments which are compared to both our line and circle models. If one or both of the segments have acceptable fits, the corresponding curve segments are marked as explained. Otherwise, the segments are left to be explained by additional critical points and the partitions they imply. The addition of critical points continues until the complete curve is explained. Figure 5 shows an example of the operation of this algorithm.

While admittedly operating in a relatively simple environment, the above algorithm exhibits excellent performance. This is true even in the difficult case of finding partition points along the smooth interface between a straight line and a circle to which the line is tangent.

Both basic principles, stability and complete explanation, are deeply embedded in this algorithm. Retaining only those partition points which persist under different "viewpoints" was motivated by the principle of stability. Our technique for evaluating the fit of the segment of a curve between two partition points, to both the line and circle models, requires that the deviations from an acceptable model have the characteristics of "white" (random) noise; this is an instantiation of the principle of complete explanation, and is based on our previous work presented in Bolles [1982].

VI DISCUSSION

We can summarize our key points as follows:

* The partition problem does not have a unique definition, but is parameterized with respect to such items as purpose, data representation, trade-off between different error types (false-alarms vs misses), etc.

* Psychologically acceptable partitions are associated with an implied explanation that must satisfy criteria for accuracy, complexity, and believability. These criteria can be formulated in terms of a set of principles, which, in turn, can guide the construction of effective partitioning algorithms (i.e., they provide necessary conditions).

One implication contained in these observations is that a purely mathematical definition of "intrinsic structure" (i.e., a definition justified solely by appeal to mathematical criteria or principles) cannot, by itself, be sufficiently selective to serve as a basis for duplicating human performance in the partitioning task; generic partitioning (i.e., partitioning in the absence of semantic content) is based on psychological "laws" and physiological mechanisms, as well as on correlations embedded in the data.

In this paper we have looked at a very limited subset of the class of all scene partitioning problems; nevertheless, it is interesting to speculate on how the human performs so effectively in the broader domain of interpreting single images of natural scenes. The speed of response in the humans ability to interpret a sequence of images of dissimilar scenes makes it highly questionable that there is some mechanism by which he simultaneously matches all his semantic primitives against the imaged data, even if we assume that some independent process has already presented him with a "camera model" that resolves some of the uncertainties in image scale, orientation, and projective distortion. How does the human index into the large semantic data base to find the appropriate models for the scene at hand?

Consider the following paradigm: first a set of coherent components is recovered from the image on the basis of very general (but parameterized) clustering criteria of the type described earlier; next, a relatively small set of semantic models, which are components of many of the objects in the complete semantic vocabulary, are matched against the extracted clusters; successful matches are then used to index into the full data base and the corresponding entries are matched against both the extracted clusters and adjacent scene components; these additional successful matches will now trigger both iconic and symbolic associations that result in further matching possibilities as well as perceptual hypotheses that organize large portions of the image into coherent structures (gestalt phenomena).

If this paradigm is valid, then, even though much of the perceptual process would depend on an individual's personal experience and immediate goals, we might still expect "hard wired" algorithms (genetically programmed, but with adjustable parameters) to be employed in the initial partitioning steps.

In this paper, we have attempted to give computational definitions to some of the organizing criteria needed to approach human level performance in the partitioning task. However, we believe that our more important contribution has been the explicit formulation of a set of principles that we assert must be satisfied by any effective procedure for perceptual grouping.

REFERENCES

1. Attneave, F., "Some Aspects of Visual Perception," Psychol. Rev., Vol. 61, pp. 183-193 (1954).

2. Bolles, R.C., M.A. Fischler, "A RANSAC-based Approach to Model Fitting and Its Application to Finding Cylinders in Range Data," in Proc. of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, B.C., Canada, pp. 637-643 (August 1982).

3. Davis, L.S., "Understanding Shape: Angles and Sides," IEEE Transactions on Computers, Vol. C-26, pp. 236-242 (March 1977).

4. Freeman, H., L.S. Davis, "A Corner-finding Algorithm for Chain-Coded Curves," IEEE Transactions on Computers, Vol. C-26, pp. 297-303 (March 1977).

5. Hoffman, D.D., W.A. Richards, "Representing Smooth Plane Curves for Recognition: Implications for Figure-Cround Reversal," in Proc. of the Second National Conference on Artificial Intelligence, Pittsburgh, Pennsylvania, pp. 5-8 (August 1982).

6. Kruse, B., C.V.K. Rao, "A Matched Filtering Technique for Corner Detection," in Proc. of the Fourth International Joint Conference on Pattern Recognition, Kyoto, Japan, pp. 642-644 (November 1978).

7. Liao, Y., "A Two-Stage Method of Fitting Conic Arcs and Straight-Line Segments to Digitized Contours," in Proc. of the Pattern Recognition and Image Processing Conference, Dallas, Texas, pp. 224-229 (August 1981).

8. Lowe, D.G., T.G. Binford, "Segmentation and Aggregation; an Approach to Figure-Cround phenomena," Proceedings of the Image Understanding Workshop, Stanford University, Stanford, California (September 1982).

9. Montanari, U., "A Note on Minimal Length Polygonal Approximation to a Digitized Contour," Communications of the ACM, Vol. 13, pp. 41-47 (January 1970).

10. Pavlidis, T., "Algorithms for Shape Analysis of Contours and Waveforms," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, pp. 301-312 (July 1980).

11. Pavlidis, T., S.L. Horowitz, "Segmentation of Plane Curves," IEEE Transactions on Computers, Vol. C-23, pp. 860-870 (August 1974).

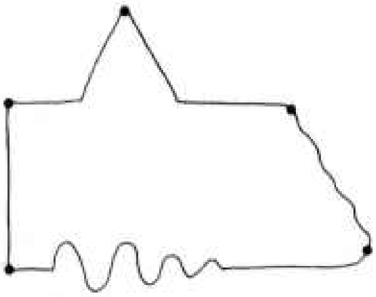
12. Ramer, U., "An Iterative Procedure for the Polygonal Approximation of Plane Curves," Computer Graphics and Image Processing, Vol. 1, pp. 244-256 (1972).

13. Rosenfeld, A., J.S. Weszka, "An Improved Method of Angle Detection on Digital Curves," IEEE Transactions on Computers, Vol. C-24, pp. 940-941 (September 1975).

14. Witkin, A., "Scale-Dependent Qualitative Signal Description," (in preparation, 1983).

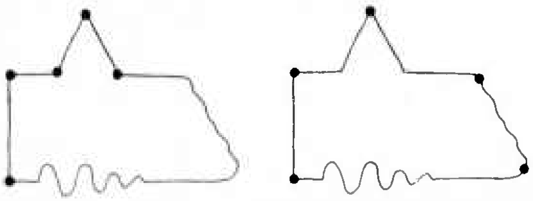
TASK 1: Select *AT MOST* 5 points to describe this line drawing so that you will be able to reconstruct it as well as possible 10 years from now, given just the sequence of selected points.

Since five points were sufficient to form an approximate convex hull of the figure, virtually everyone did so, selecting the 5 points shown below.



TASK 2: Assume that a friend of yours is going to be asked to recognize this line drawing on the basis of the information you supply him about it. He will be presented with a set of drawings, one of which will be a rotated and scaled version of this curve. You are only allowed to provided him with *A SEQUENCE OF AT MOST 5 POINTS.* Mark the points you would select.

Since 5 points were not enough to outline all the key features of the figure, the subjects had to decide what to leave out. They seemed to adopt one of two general strategies: (a) use the limited number of points to describe one distinct feature well (illustrated by the selection on the left), or (b) use the points to outline the basic shape of the figure (shown on the right).



TASK 3: This line drawing was constructed by piecing together segments produced by different processes. Please indicate where you think the junctions between segments occur *AND VERY BRIEFLY DESCRIBE EACH SEGMENT.* Use as few points as possible, but no more than 5.

The constraint of being limited to 5 points forced the subjects to consider the whole curve and develop a consistent, global explanation. The basic strategy seemed to be a recursive one in which they first partitioned the curve into 2 segments by placing a breakpoint at position 1 and another one at either position 2 or position 3 to separate the smooth curves from the sharp corners. Then they used the remaining points to subdivide these segments according to a vocabulary they selected that included such things as triangles, rectangles, and sinusoids. For example, almost everyone placed breakpoints at positions 3 and 4 and described the enclosed segment as part of a triangle. Similarly the segment between positions 1 and 5 was generally described as a decaying sinusoid. It is interesting to note that in task 1 the subjects consistently placed a point close to position 5 but always farther to the right, because they were trying to approximate a convex hull. The different purposes led to different placements.

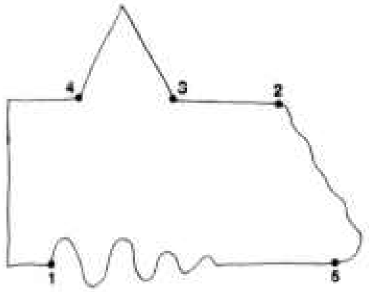
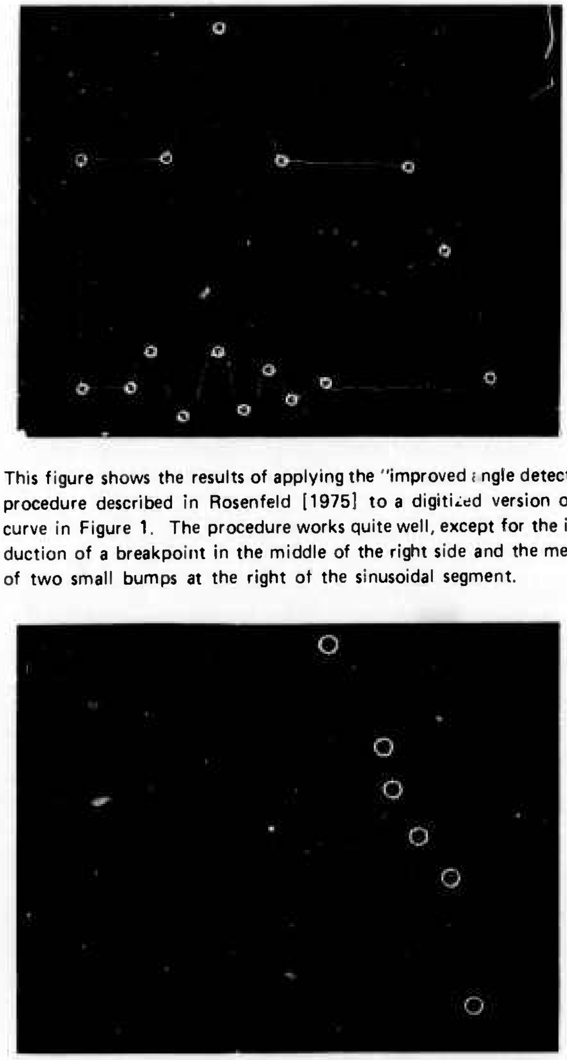


FIGURE 1 EXPERIMENTS IN WHICH HUMAN SUBJECTS WERE ASKED TO SEGMENT A CURVE



(a) This figure shows the results of applying the "improved angle detection" procedure described in Rosenfeld [1975] to a digitized version of the curve in Figure 1. The procedure works quite well, except for the introduction of a breakpoint in the middle of the right side and the merging of two small bumps at the right of the sinusoidal segment.

(b) However, if we extract a portion of the curve and apply the algorithm, it introduces several additional breakpoints because the change in curve length causes some of the algorithm parameters to change.

FIGURE 2 ESTIMATION OF CURVATURE FROM DISCRETE APPROXIMATIONS

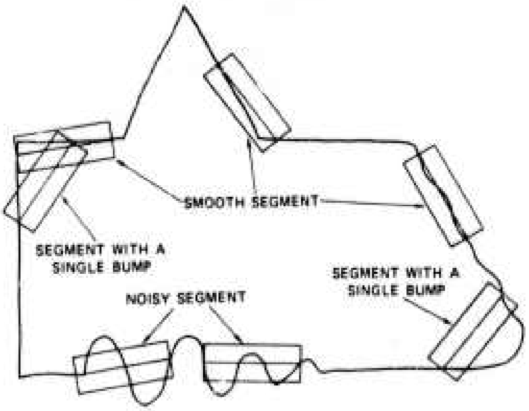
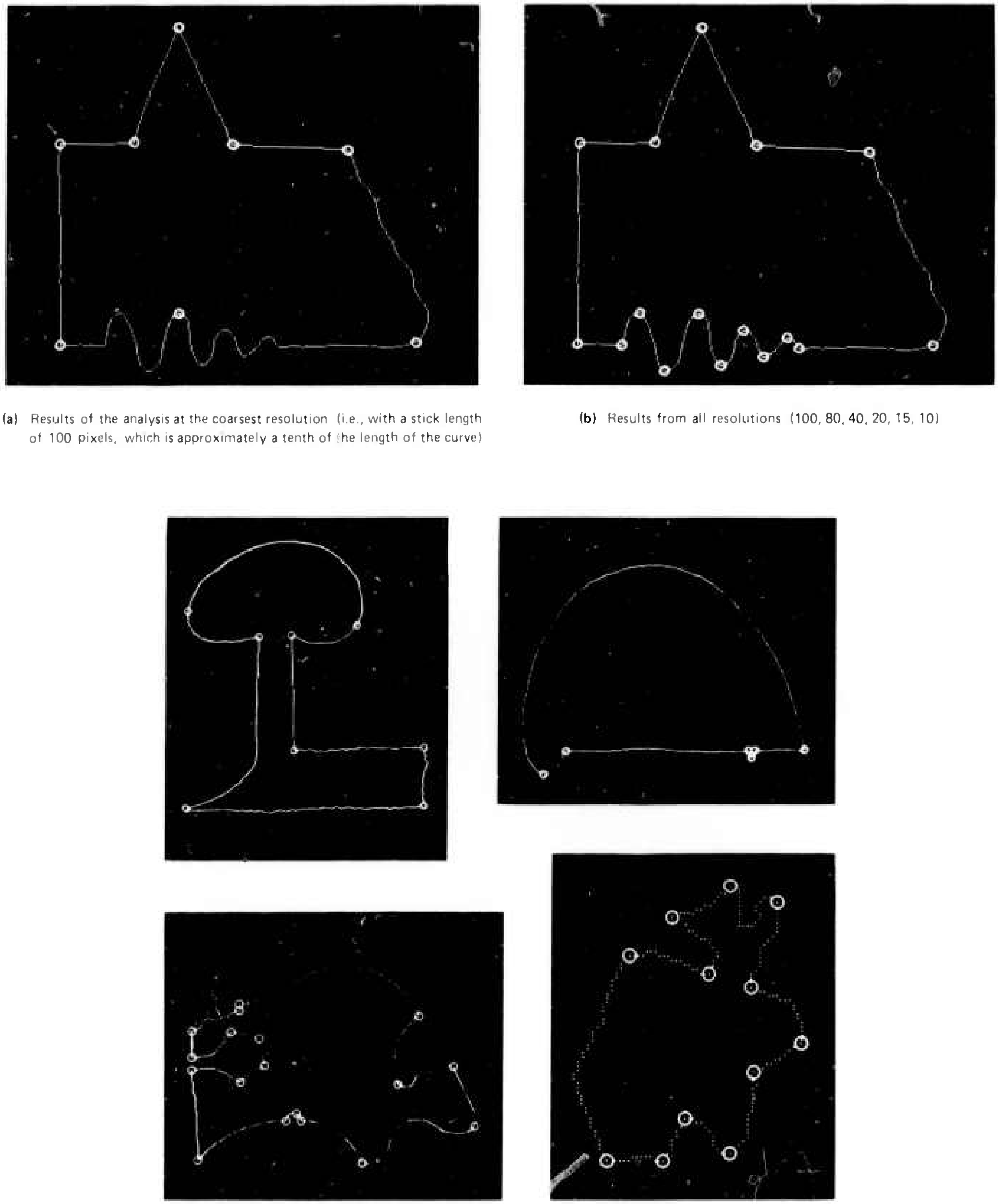


FIGURE 3 EXAMPLE CURVE SEGMENTS AND THEIR CLASSIFICATIONS

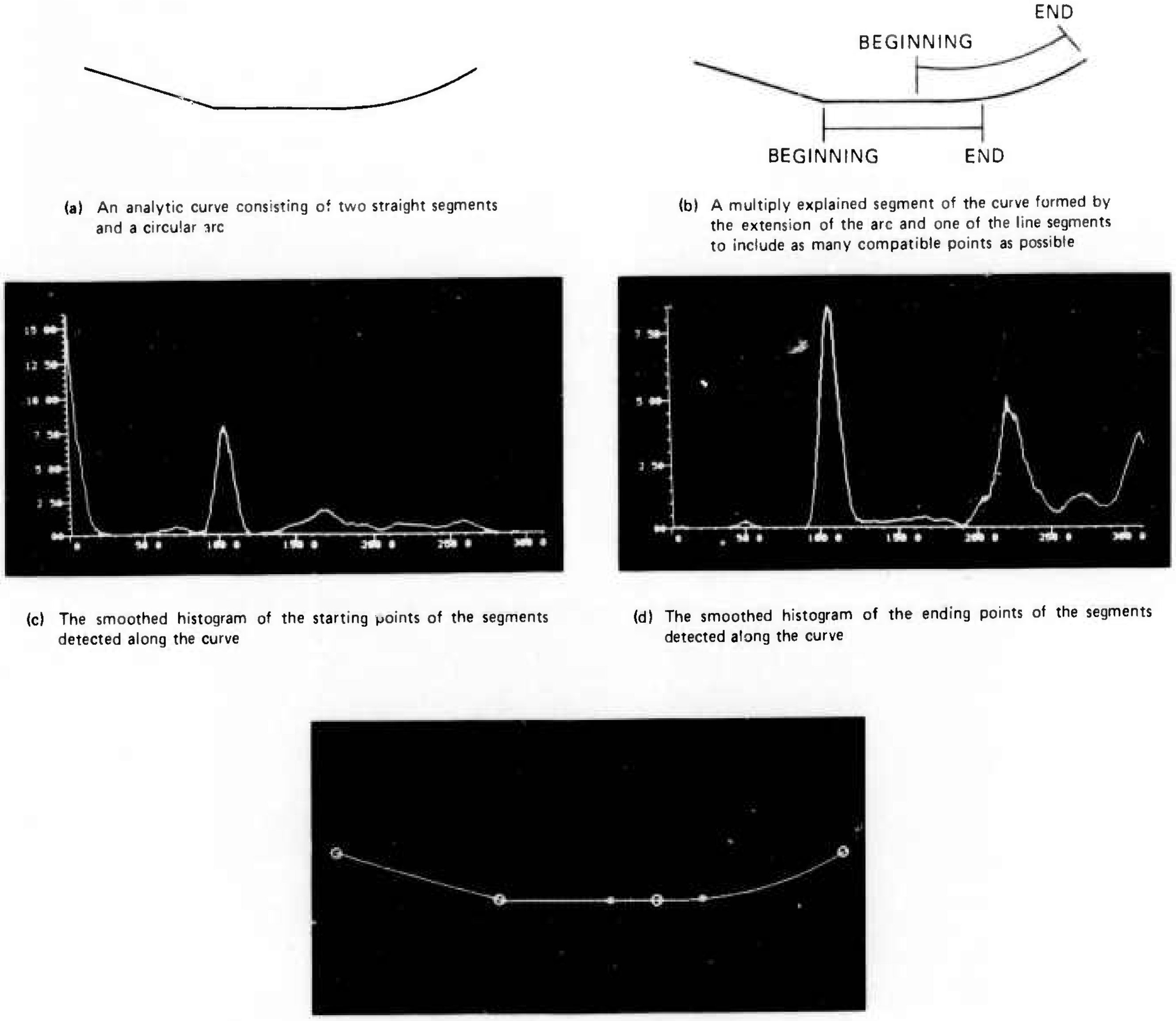


(a) Results of the analysis at the coarsest resolution (i.e., with a stick length of 100 pixels, which is approximately a tenth of the length of the curve)

(b) Results from all resolutions (100, 80, 40, 20, 15, 10)

(c) Additional examples of the local discontinuity analysis

FIGURE 4 LOCAL DISCONTINUITY PARTITIONING



(a) An analytic curve consisting of two straight segments and a circular arc

(b) A multiply explained segment of the curve formed by the extension of the arc and one of the line segments to include as many compatible points as possible

(c) The smoothed histogram of the starting points of the segments detected along the curve

(d) The smoothed histogram of the ending points of the segments detected along the curve

(e) The breakpoints suggested by the histograms. The breakpoint between the arc and the line was placed at the center of the mul- multiply explained region, which is bounded by asterisks.

FIGURE 5 PROCESS PARTIONING

Scene Matching by Hierarchical Correlation

Frank Glazer

Corporate Research & Architecture, Digital Equipment Corporation
77 Reed Rd., Hudson, Massachusetts

George Reynolds P. Anandan

Computer and Information Science Department[1]
University of Massachusetts at Amherst

Abstract

In this paper ~~we~~ the authors present an implementation of hierarchical scene matching in the VISIONS image processing cone - a pyramidal processing architecture. The problem of scene matching is common to many applications in machine vision including registration, motion detection, and stereo vision. Scene matching by feature correlation can solve this problem but suffers from computational expense and failure in highly textured images. Hierarchical correlation provides both a cheaper matching algorithm and a coarse-to-fine matching strategy that overcomes "textural" problems by matching on gross image structures first. These methods fit naturally into the processing cone or pyramid architectures that have been proposed for image processing. We present a discussion of the architecture of the processing cone, the construction of image pyramids, and the use of these pyramids in hierarchical correlation. A set of experiments illustrates the operation of these ideas.

Presented is

0.0 Introduction

The problem of matching digital images by correlation techniques is an important and well known problem in computer vision and pattern recognition. It has applications in image registration, object detection by template matching, and motion and stereo analysis. This paper describes a hierarchical approach to this problem, and includes some results on noisy real world images.

While there have been a number of studies of correlation techiques for matching images [1,14], the two basic problems regarding computational costs and false match have not been solved. If the image displacements are limited to be less than D pixels in either direction, then there are $(2D + 1)^2$ possible test locations for matching at each pixel. The problem of false matches may arise for several reasons. High frequency texture in the image may provide enough repetitive pattern that a number of different matches may be considered equally valid by the correlation technique. On the other hand the lowest frequencies may be due to illumination differences and may bias the correlation measure away from the veridical match. Different kinds of normalizations can help overcome this problem, but only at greater computational cost [6]. In order to avoid false matches it may be necessary to use large sample windows, which also increases this cost.

The hierarchical matching technique described here overcomes both of these problems. First, the matching is done initially based on the larger structures in the images (since they become prominent at low frequencies), providing ball-park estimates for matching higher frequency information at levels below. This overcomes the problems due to high frequency textures. Secondly, the coarse-fine strategy restricts the search to 3 x 3 areas at each level significantly reducing the computational cost.

Basically the technique consists of matching band-passed versions of the images at different levels of resolution. The filters applied approximate convolution with ∇^2G operators of different sizes.[1] The size of the Gaussian increases as the resolution becomes coarser, in such a way as to limit the frequency content in the image to avoid aliasing due to the sampling rate at each level of resolution. The elimination of low-frequencies in the image helps overcome any problems due to illumination and scaling differences.

While this sort of approach to improving matching has been discussed by other authors [15,17,12,9,3], to our knowledge only Wong and Hall have applied it to real images and studied the issues in doing so. Our technique, which more closely resembles the one outlined by Burt [3], differs significantly from the approach of Wong and Hall.

[1] This research was supported in part by DARPA under Grant N00014-82-K-0464.

[1] The ∇^2G (read del-two-g) operator is the Laplacian applied to the result of convolving with a Gaussian.

1.0 The Processing Cone Structure and Image Pyramids

In our hierarchical algorithm images are represented at varying levels of spatial resolution. Coarse resolution images are obtained by low pass filtering and sub-sampling. Normally the filtering is done by convolution smoothing with Gaussian-like kernels. This low-pass filtering allows us to sub-sample these images and store them in coarse grids

1.1 The processing cone

The natural architecture for these image pyramids is the processing cone [8], a multilayer, multiresolution organization of image planes upon which inter- and intra-layer image operators are applied (see Figure 1). Operations which produce coarse images from finer ones are called *reductions*, while those that produce finer resolution images from coarse ones are called *projections*. This hierarchical data structure has also appeared as *pyramids* in [16]. Similar multiresolution representations have been used before in scene matching [17,13,12].

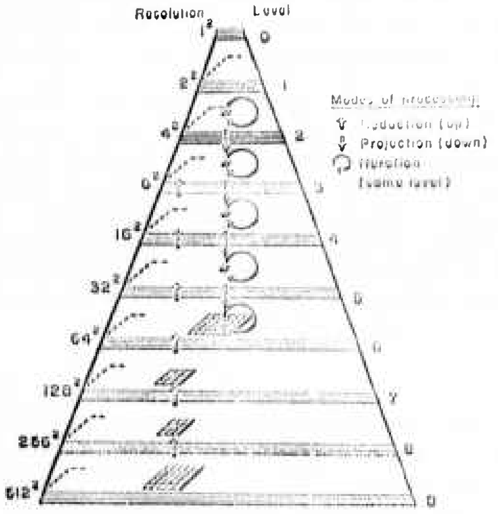


Figure 1: The processing cone.

This parallel array computer is hierarchically organized into layers of decreasing spatial resolution. Information within the cone is transformed by means of functions operating on local windows of data. Cone algorithms are specified as sequences of these parallel functions applied in one of three processing modes: reduction (up the cone), projection (down the cone), and iteration (at the same level).

The processing cone is composed of levels 0 to L, each level being 2^L pixels on a side. When we place two adjacent levels in registration, each coarse pixel overlays four finer pixels. We will call these pixels *fathers* and *sons* respectively. These pixels cover the same square area of the image space. Going from one level to a coarser we get a four to one reduction in data. The highest spatial frequency that can be represented at a coarse level (corresponding to the Nyquist rate) is half that which can be represented at the next finer level. Thus for each step up the cone (coarsening) the spatial frequency bandwidth (relative to the finest grid) is cut in half.

1.2 Low pass pyramids

When we fill the cone with reduced resolution copies of the image by subsampling, low pass filtering must be done to prevent aliasing. Aliasing occurs when the one-of-four subsampling that gives the next coarser level produces spurious image components for any spatial frequency in the upper halves of the frequency spectrum of the image being sampled. The simplest low pass filter we can use is obtained by taking the average of the four sons of a coarse pixel (as in [16] and [17]).

A slightly more complicated family of reductions has been proposed by Burt [2]. He has approached the inter-level low-pass filter design by considering the net convolution obtained when a given reduction is applied at progressively higher (coarser) levels. For example, the two-by-two averaging reduction when applied twice is equivalent to a four-by-four averaging reduction up two levels in the cone. Continued application of two-by-two averaging up the cone always results in "flat" equivalent convolution masks (i.e. unweighted averaging). Burt shows how by using slightly larger four-by-four kernels the equivalent convolution masks can be made to approximate Gaussian-like low-pass filters. We have used the following such kernel in most of our experiments [1 3 3 1] x [1 3 3 1]^t.[1] The result of applying this 4 x 4 operator on an image appears in Figure 2.

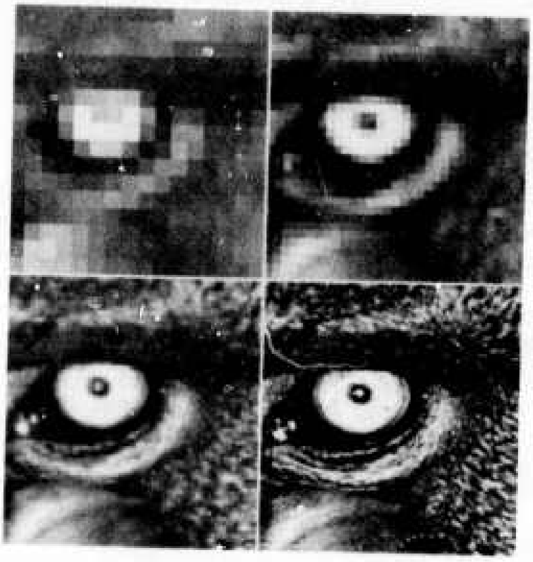


Figure 2: Low pass pyramid.

Levels 4 through 7 of the low pass pyramid obtained from the mandrill eye image by applying the 4 x 4 reduction operator [1 3 3 1] x [1 3 3 1]^t.

[1] [. . .] is a column vector, 'x' is the outer product operation, and 't' is the transpose operator.

1.3 Band pass pyramids

In the next section we will see that correlation matching is better performed when low spatial frequencies (relative to the grid size) have been filtered out. Such high-pass filtering performed at each level of a low pass pyramid effectively produces a band-passed image at each level.

A good choice for this high-pass filtering is a discrete Laplacian such as $\begin{matrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{matrix}$ or $\begin{matrix} 1/4 & 1/2 & 1/4 \\ 1/2 & -3 & 1/2 \\ 1/4 & 1/2 & 1/4 \end{matrix}$. Such a mask can be applied to each level of the low-pass pyramid.

A second method for generating a band-pass is Burt's Laplacian pyramid [3,5]. Here the fact that a $\nabla^2 G$ can be approximated by a difference of Gaussians is used to effectively compute band-pass filters by differencing adjacent levels of a Gaussian (low-pass) pyramid. The difference is taken between the finer level and an appropriate projection of the coarser level. Figure 3 shows the Laplacian pyramid derived from the Gaussian pyramid in Figure 2.

Finally, a third method for computing a band-pass pyramid, is to perform a Laplacian at the finest level and then use the high-pass output as the base of a low-pass pyramid. This method is used in the optic fundus image experiments in section 5.2 . Some of our recent theoretical work has suggested that this method has an aliasing problem, but we have not yet seen it in our experiments.

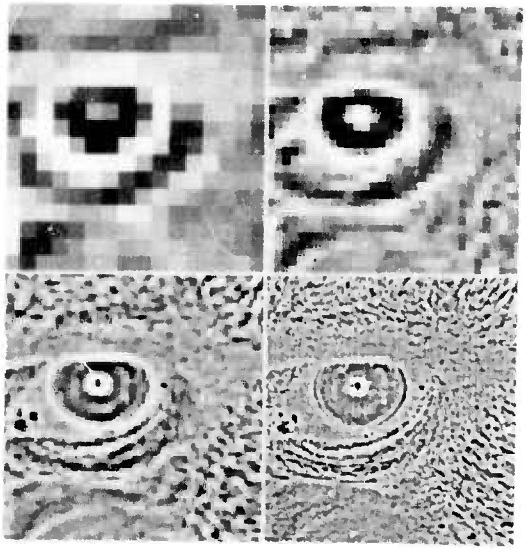


Figure 3: Band pass pyramid.

Levels 4 through 7 of the band pass pyramid obtained from the low pass pyramid in Figure 2.

2.0 Correlation Matching

In correlation matching a *sample window* about a point in one image is compared to trial windows in the second image. The point in the second image whose trial window gives the optimal correlation value is chosen as the match point. This can be considered a special case of the more general method of feature matching. An example of alternative features are the edges that were used in Marr and Poggio's matching algorithm [12].

There are more than a few variations of the basic correlation measure [e.g., see 6]. The *basic correlation* from which the family of measures derives its name is the sum of the pairwise product of corresponding pixels in two windows. Common variations include 1) mean normalized correlation, in which the mean of the values in each window is subtracted from each value; 2) variance normalized correlation, in which the correlation sum is divided by the variance of the two windows; 3) sum of squared differences, and 4) sum of the magnitude of differences.

Mean normalized correlation is equivalent to basic correlation performed after a specific pre-filtering of the two images. The filter (or convolution mask) used in this case is the difference of a "flat" averaging mask and the identity mask (a discrete impulse). As such it can be considered as a high-pass filter and is thus related to other high pass filters such as the discrete Laplacian. However, the frequency response of the subtract-local-mean filter is not as flat at high frequencies as that of discrete Laplacians. For this reason, we are led to consider Laplacian pre-filtered basic correlation as a substitute for mean normalized correlation.

In our experiments we have used 8 x 8 sample windows. This choice is intended to provide a tradeoff betweem small windows which are more immune to occlusion and distortion problems and large windows which capture a large amount of matchable structure. We have not yet experimented with other sample window sizes.

3.0 Search Strategy

One way of looking at matching is as a process of searching for the point that optimizes some measure of similarity. In our case the measure is the local correlation measure between the two images. The strategy adopted for searching for the point of match (i.e., the point where the measure is maximized) should not only attempt to decrease the number of false matches, but also reduce the computational cost involved in searching.

3.1 3 by 3 Search

The search strategy adopted in our process begins at a coarse level where the maximum displacement is within one pixel in both directions (see Figure 4). Let this be level L_d. The search is conducted in a 3 x 3 area around the point of interest at level L_d in the band-pass images at that level. The resulting displacements at this level (along each axis) are either -1, 0, or 1. The value obtained here is within 1/2 pixel of the correct displacement at this level.

At each level below (say L_i), the displacement values for a given point are projected down from its father pixel in the next level above. Due to the doubling of resolution (thus halving the pixel width) this value is double the value of the father pixel. This establishes the displacements within one pixel accuracy in either direction at this level (L_i). Searching in a 3 x 3 area at this level refines the displacement to within 1/2 pixel accuracy at this level. The process is repeated up to and including the finest level of resolution of the image.

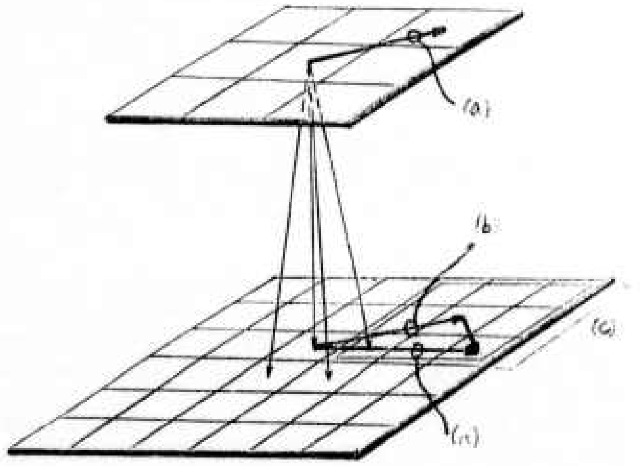


Figure 4: 3 by 3 search.

(a) Displacement vector at level N
(b) Displacement vector projected to its four sons
at level N+1 (only one of the four sons is shown)
(c) Search in a 3 x 3 area at level N + 1
(search area shown in double lines)
(d) Updated displacement vector

3.2 Computational Costs

The computational advantages of hierarchical versus single level search strategies can be measured in two ways. We can consider how many points are searched in arriving at a final match at the finest level. Each ancestor of the final point matched will contribute to this measure. On the other hand, we can measure the cost of obtaining matches at all points at the finest level. The former measure should be used when matching is confined to a relative sparse set of interesting points, while the latter is used when matching is done almost everywhere. Since each of these case are interesting, we will look at both.

First, let us consider the comparative cost of arriving at a single match at the finest level. Let D be the maximum displacement (measured at the finest level). Then the initial coarse search will be performed $\log_2 D$ levels above the finest level. The number of points searched is $(\log_2 D + 1) * 9$, because the search at each level is restricted to a 3 x 3 neighborhood and there are $\log_2 D + 1$ levels for search. On the other hand, a correlation process searching at one level through all points closer than the maximum displacement uses $(2D + 1)^2$ points for comparison.

Now consider the cost of computing matches at all points in the finest level image. Let the finest level image contain N^2 points and, as above, let D be the maximum displacement. Single level correlation would then search $N^2(2D + 1)^2$ points. Hierarchical correlation would search $9 * (N^2 + N^2/4 + N^2/16 + ...)$ points, where the summation is over all levels at which matching takes place. However high those levels go, the sum is always less than that of the corresponding geometric sequence, viz. $4N^2/3$. Thus the number of points searched hierarchically is $12N^2$. Table 1 shows a comparison of costs.

D	S	H	S/H	S/12
1	9	9	1.0	—
2	25	18	1.4	2.1
4	81	27	3.0	6.75
8	289	36	8.0	24.1
16	1089	45	24.2	90.1
32	4225	54	78.2	352.

Table 1: Cost of single level vs. hierarchical search.

This table compares hierarchical and single level search strategies. D is the maximum displacement, $S = (2D+1)^2$ is the cost of single level search, $H = 9(\log_2 D + 1)$ is the cost of *single match* hierarchical search, S/H is their relative cost factor. S/12 is the relative cost factor between single level and hierarchical *full matching* (i.e. at all points).

3.3 Existence and Uniqueness of Matches

The discussion in section 3.1 shows how the technique of coarse-fine search strategy automatically ensures that the correct match must exist within the 3 x 3 local search window at each level. The filtering and subsampling processes ensure that the highest frequency at a particular level corresponds to a wavelength of two pixels at that level. Since the search is restricted to 3 x 3 windows at that level, we have some confidence that the match obtained within this window is unique. Also, the elimination of lower frequencies at that level help provide sufficient variation in the correlation measure within this window.

Strictly speaking, this argument is valid only for a one dimensional version of this process. In the two dimensional case there is no high frequency content along straight edges. This can lead to nearly constant values of the correlation measure along these edges, thus leading to false matches. This, in fact, leads us to the use of interest operators to eliminate points that can potentially lead to false matches.

4.0 The Problem of False Matches

The match for a point obtained by the correlation technique may not always correspond to it's environmental match. There are three basic reasons for this. First, correlation only deals with translational disparity (in the image plane) and so should break down with increasing rotational and scaling components in the disparity. However, as the optic fundus image experiments show, small rotations can be dealt with. Secondly, in practical imaging situations there can be significant amounts of noise in the images. Most often this would most adversly affect matching at finer resolutions. A third cause of false matches is the occurence of occlusions in an image. Two problems can arise here 1) points in one image may have no counterpart in the other image; 2) points on an occlusion boundary have neighboring windows which change identity from frame to frame.

4.1 Interest operators

Interest operators are designed to pick out points for which matches can be found with a high reliability. This detection of *matchability* is the key element of an interest operator. They can also be used to restrict processing to a small subset of all the image points to reduce computation costs. On serial machines this is certainly useful. However, on image parallel machines such as the processing cone, we need only be concerned with matchability.

In order that a point in one image image be matchable in another image, the point must be matchable with itself. For this to be true the local autocorrelation function must possess a strict local maxima. A sufficient condition for this is the occurrence of a strong corner at a point. Kitchen and Rosenfeld [11] present an analysis of various corner finding algorithms and these algorithms yield very good interest operators. Moravec [13] gives an interest operator which attempts to compute the sharpness of an approximate autocorrelation function directly.

In our hierarchical correlation experiments we have taken two approaches to applying interest operators. In the first approach, an interest operator is applied at the finest level of the first frame to select the points to be matched, and then a logical pyramid is formed by using "OR" in the 4 x 4 reduction operation. Matching is only performed at those points which have a value of TRUE in this pyramid. This method is comparable to Moravec's search strategy [13].

In the second approach, interest operators are applied at all levels. In this case there can be interesting pixels with un-interesting fathers. For these pixels, we can do one of two things: 1) the search can be done in a larger search area, or 2) a displacement estimate can be obtained based on neighboring pixels.

We are currently studying both approaches but we only present the first approach in the experiments below. One of the surprising results of our experiments is that even at points which appear to be "uninteresting", correct matches are obtained. Depending on the domain of application, our experiments show that a large amount of computation to find interesting points may be unnecessary.

5.0 Experiments

5.1 Mandrill image experiments

In this experiment we took the standard USC image of a mandrill and extracted a 128^2 subimage of it (Figure 5a). We created a second image by adding white gaussian noise to this image and translating it 5 pixels up and 7 pixels to the right with respect to the first image (Figure 5b). The standard deviation of the noise added was 25.0 which is 10% of the intensity range of the image.

We conducted two experiments with these images. The first was the hierarchical matching process. The Laplacian pyramids were constructed using Burt's techniques. The matching at each level was done using an 8 x 8 sample window at all points in the image. The results at the various levels are shown in Figure 6 and Figure 7. Figure 6 shows results at levels 4,5,6, and 7. At each level the displacement estimates are shown at a sampling of 64 points.

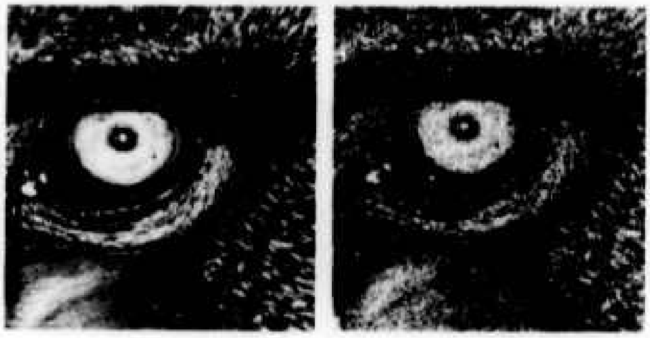


Figure 5: Mandrill eye images used in the first experiment.

(a) A 128^2 piece of the larger mandrill image.
(b) A similar piece, translated 5 pixels up and 7 to the right, with white gaussian noise added (standard deviation = 10% of full range).

In Figure 7 two-dimensional histograms of the row versus the column components of the displacements are shown for each of level 4 through 7 (Figure 7, a through d). Note, in Figure 7d, the high count found in the bucket corresponding to the correct displacement of (-5,7). The histogram for level 7 (Figure 7d) indicates that about 87% of the displacement values are exact. This shows that the hierarchical process is quite insensitive to noise.

In the second experiment, we attempted to match these two images using a correlation process all at one level. In doing this we used 8 x 8 sample windows and searched in a 17 x 17 search area around each pixel (the actual displacements of -5,7 will fall within this range). The results of this process are shown in Figure 8. Note the greatly reduced accuracy of this method (53% correct).

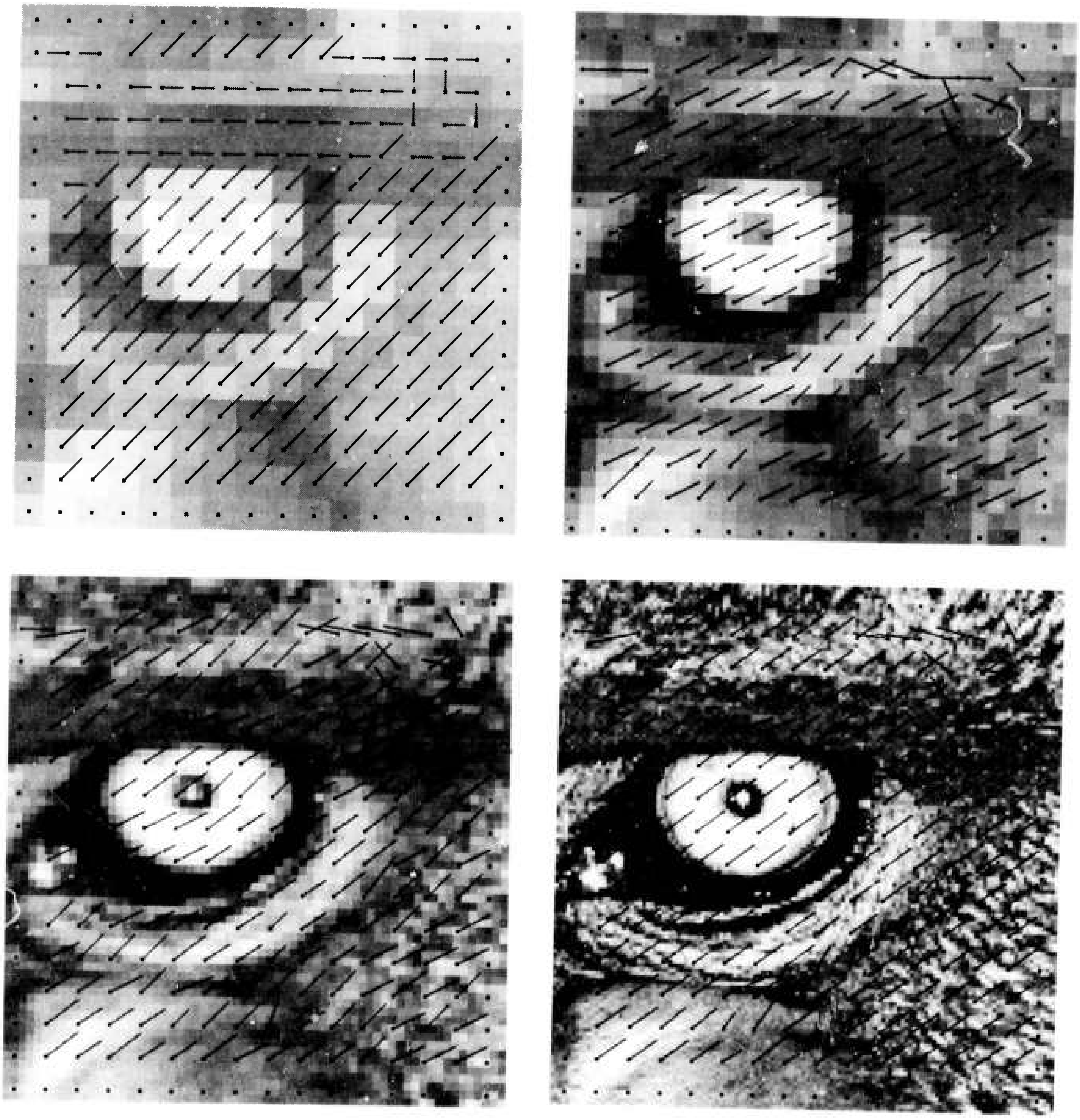


Figure 6: Computed displacement vectors.

The displacement vectors at levels 4 through 7 obtained in the Mandrill experiment. Only a 64^2 sample of vectors is shown at each level.

(a)

	-1	0	1
-1	0	4	148
0	8	1	35
1	0	0	0

(b)

		-2		0		2	
	0	0	0	0	0	0	0
-2	0	0	4	0	0	31	0
	9	0	7	0	153	527	1
0	14	0	0	0	0	0	0
	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
	0	0	0	0	0	0	0

(c)

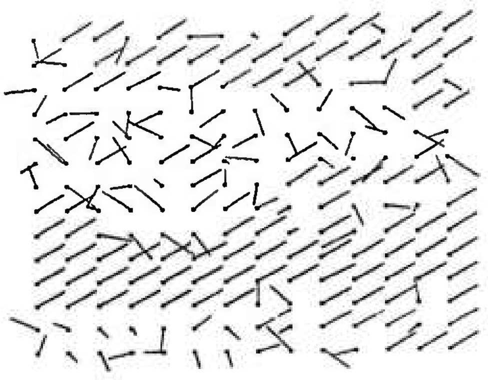
		-6		-4		-2		0		2		4		6	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
-4	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0
	1	3	0	0	3	10	7	0	4	2	708	359	31	0	0
-2	4	1	3	0	5	0	0	0	5	58	909	701	62	2	0
	0	43	13	0	0	1	13	0	1	0	0	0	5	0	0
0	1	4	0	0	0	0	0	0	0	0	0	0	0	0	0
	3	12	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(d)

		-14		-12		-10		-8		-6		-4		-2		0		2		4		6		8		10		12		14	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
-8	0	0	0	0	0	0	0	0	2	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	4	0	0	0	0	0	3	3	23	9	4	1	0	1	0	5	0	11	12	8	1	12	1	0	0	0	0	0
-6	0	4	3	4	0	0	0	0	4	0	0	5	4	17	0	0	0	0	7	5	4	12	14	7	9	9	40	0	0	0	0
	8	1	0	4	1	0	0	0	14	2	0	0	5	0	0	0	3	0	39	7	30	57	10322	57	74	90	19	3	0	0	0
-4	6	0	0	0	1	2	1	0	1	6	0	0	0	0	0	0	0	9	56	51	58	7	4	1	35	31	25	0	5	0	0
	0	0	8	26	51	9	8	0	0	4	3	0	1	9	2	0	11	0	8	22	30	9	24	18	12	27	47	0	0	0	0
-2	0	0	3	27	6	0	0	0	0	0	0	0	10	3	7	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0
	0	0	1	38	45	6	5	0	0	0	0	0	14	2	6	0	4	0	0	0	0	0	0	0	0	0	3	0	0	0	0
0	0	0	3	10	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	3	0	24	6	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7: Distribution of displacement vectors.

The histograms of the row and column components of the displacements obtained in the Mandrill experiment. Levels 4 through 7 have been shown. Note the peak at (-5,7) in figure 7d.



	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8
-8	29	6	22	21	18	8	36	28	23	10	25	13	26	17	28	13	15
-7	8	5	10	51	19	8	53	28	10	7	7	16	13	20	6	13	16
-6	22	50	15	50	24	3	13	7	6	30	16	19	9	23	42	75	39
-5	11	18	19	21	15	2	29	4	18	22	5	33	37	20	72	6279	138
-4	3	17	10	17	11	12	17	1	36	5	8	10	13	22	29	16	64
-3	32	34	11	6	10	33	17	31	24	12	24	19	17	37	41	15	1
-2	4	8	31	24	49	26	39	5	23	9	8	52	3	9	9	29	11
-1	17	17	2	15	22	42	16	29	22	29	29	3	6	21	2	11	10
0	29	13	13	31	25	21	19	37	3	17	40	10	7	7	7	39	19
1	3	6	46	4	43	6	6	44	4	14	34	17	1	11	11	30	8
2	7	2	30	44	14	24	52	21	5	27	26	27	27	17	4	21	33
3	3	32	19	9	15	21	23	26	35	32	31	26	23	20	17	20	20
4	7	19	54	7	20	21	18	28	18	12	5	8	45	21	53	3	33
5	34	20	3	11	13	17	3	17	4	31	13	13	57	20	11	40	7
6	8	4	57	16	4	11	13	4	12	14	35	15	30	9	24	7	39
7	24	28	11	27	17	129	105	28	3	28	36	17	66	16	8	16	60
8	22	26	86	30	18	33	6	42	21	22	21	5	47	19	21	14	17

Figure 8: Single level correlation.

Results of the variance normalized correlation applied to the Mandrill images (Figure 5).
(a) shows the displacement vectors and (b) shows the displacement histograms.

5.2 Optic fundus image experiments

The next problem to which we applied the hierarchical matching algorithm was that of registering two fluorescein angiogram images of the optic fundus (see Figure 9). These images were obtained from Paul Nagin at the Tufts New England medical center and are digitized as 128^2 images. The problem is to register two images taken at the beginning and at the peak of dye filling. Areas which show very little change are recognized as regions where no filling of the dye is taking place. This measurement can then be used in the prognosis of glaucoma. Due to severe contrast changes over this time interval it is necessary to register a temporal sequence of 8 to 10 images.

In this experiment the finest level image was bandpass filtered using a $\nabla^2 G$ convolution. The Gaussian convolution used has a standard deviation of two pixels and introduces some smoothing at the finest level. In fact, we implement this convolution using the Fast Fourier Transform with the filtering done in the frequency domain. A pyramid is formed using the $\nabla^2 G$ filtered image as the base. In Figure 10 we show the results of applying the matching algorithm at the bottom four levels. Again, we have subsampled the vector field for display purposes.

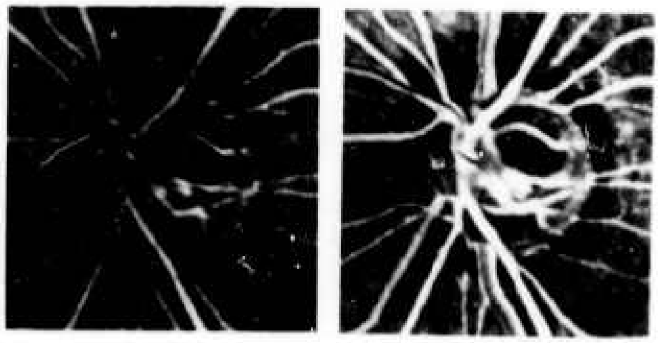


Figure 9: Optic fundus test images.

These are real images taken at two successive time instants. Note the large change in mean intensity.

For the images used in this experiment any three dimensional effects due to the movements of the eye can be safely ignored, so the misregistration is due to a rigid motion in the plane (viz., eye movements and mis-alignments in the digitization process). The problem then is to find the rigid motion which will bring the images into register.

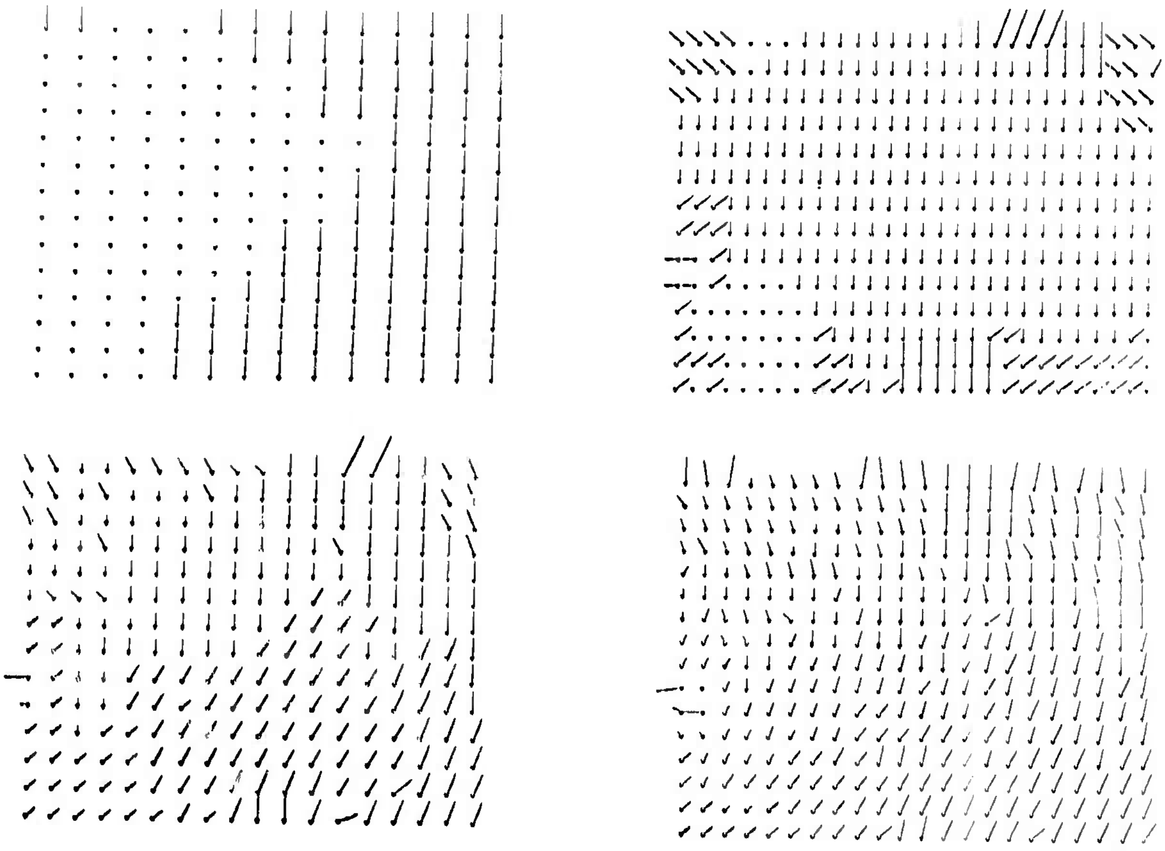


Figure 10: Computed displacement vectors.

The displacement vectors at levels 4 through 7 obtained in the optic fundus experiment. Note that the displacement fields at have a distinct rotational component.

To measure the accuracy of the matching algorithm we generated a vector field by computing the rigid transformation that best fit the computed displacement field. This field was subtracted from the displacement computed by the matching algorithm. A two dimensional histogram of the row versus the column components of the difference vectors is shown in Figure 11. Figure 12 shows the same type of histogram with the differences being taken only at a set of interesting points. In this case 76 interesting points were computed at the finest level using the Kitchen-Rosenfeld corner finder [11] and then "OR-ing" the points up the pyramid.

The central bucket of the histograms corresponds to an error of at most 1/2 pixel in either of the row or column directions. About 35 percent of the points in the histogram of Fieure 11 are in the central bucket whereas in Figure 12, approximately 55 percent of the points are in the central bucket. This indicates that better accuracy can be obtained using an interest operator. However, the concentration of points around the central bucket in Figure 11 suggests that global statistics of the displacement field can be accurately obtained without the use of an inte.est operator.

		-6		-4		-2		0		2		4		6	
	0	0	0	0	0	0	0	0	0	2	19	0	0	34	0
-6	0	0	0	0	0	0	0	0	0	2	7	2	1	0	0
	0	0	0	0	0	0	0	0	27	13	28	2	0	0	0
-4	0	0	0	0	21	0	0	0	21	1	3	0	0	0	0
	0	0	0	0	16	14	0	6	18	7	18	0	0	1	0
-2	0	0	0	0	0	74	26	67	39	60	6	1	4	2	0
	0	0	0	0	0	15	424	857	374	46	21	0	0	0	0
0	0	0	0	0	5	3	1100	4382	661	9	0	0	0	0	0
	5	2	0	8	25	51	658	1825	665	51	0	0	0	0	0
2	0	5	4	4	0	5	7	13	35	0	26	0	0	0	0
	12	1	0	0	0	2	0	1	1	0	0	0	0	0	0
4	6	0	0	0	0	0	0	0	0	7	0	0	0	0	0
	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11: Distribution of difference vectors.

The error histogram obtained by differencing the computed displacement field from the field generated by the translational and rotational paramters derived from the computed field.

		-6		-4		-2		0		2		4		6	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	4	2	2	0	0	0	0	0	0
0	0	0	0	0	0	0	5	34	8	0	0	0	0	0	0
	0	0	0	0	0	0	4	7	4	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0

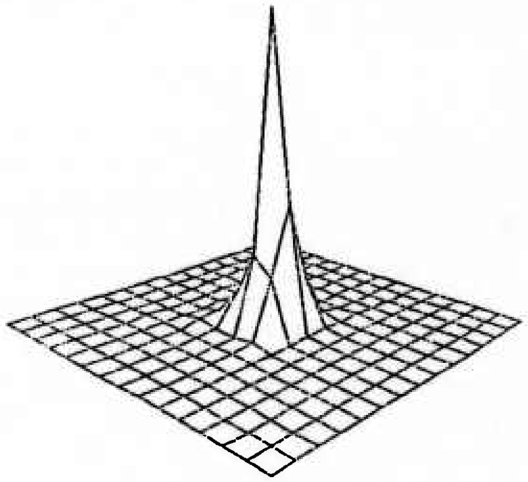
Figure 12: Distribution at interesting points.

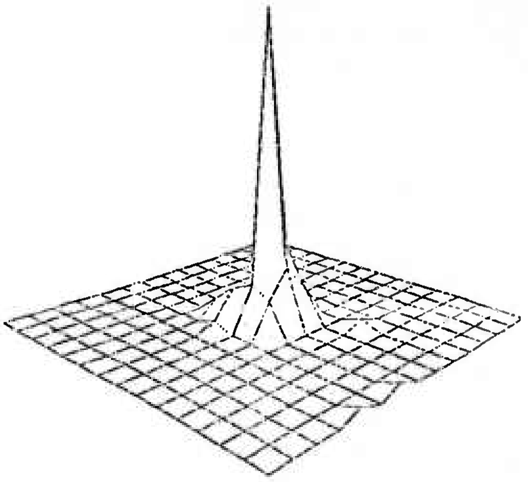
Using the same data as in previous figure, this histogram was obtained by including only the displacement vectors at interesting points.

6.0 Future Directions

Our experiments have shown that hierarchical matching provides an excellent method for the computation of displacement fields. However, a thorough evaluation of the effects of various parameters and algorithmic options on the accuracy of the computed fields has yet to be carried out. These include:

1. *The size of the sample window*; Can windows as small as 3 x 3 provide adequate matches ?
2. *The shape of the sample window*; How do center-weighted windows (e.g., Gaussian windows [6]) improve accuracy when the disparity field is not smoothly varying ?
3. *The method of computing the bandpass pyramids*; This issue has two parts: a) effect of the method on the accuracy of the displacement field; and b) efficient computational implementations
4. *The use of normalized correlation*; Bandpass filtering and the small 3 x 3 search areas seem to eliminate the need to do normalized correlation. This may not remain true if sample windows smaller than 8 x 8 are used.





While the absolute accuracy (i.e, percent correct) of the displacement field is important, the degree and nature of tolerance of errors greatly depend on its intended application. For example, in image registration applications, the statistical distribution of errors is more significant than errors at specific points. On the other hand, some of the structure from motion algorithms require highly accurate displacements at a few speicific points. Hence, a study of the **evaluation criteria** of the displacement fields with careful consideration of the application domains is an important area of future work.

Another important research problem is a systematic study of interest operators and sharpness measures. It is necessary to understand how they relate to the degree of confidence in the displacements obtained at image points. This issue is also linked to the issue of how to obtain an dense accurate displacement field from a sparse field (i.e, one computed only at points of high confidence) or from a dense, but inaccurate field (with the knowledge of the degree of confidence of the displacements). Typically, these involve applying smoothing or interpolation processes to the displacement fields. It is important to note here that there are hierarchical techniques [7] that dramatically improve the speed of some of the iterative smoothing processes. In face it appears that hierarchical matching and interpolation can be performed together.

7.0 Summary

In this paper we have described the implementation of a hierarchical correlation process in the processing cone architecture of the VISIONS Image Operating System. Two representative experiments were presented to describe its performance. The hierarchical process involves matching band-pass filtered images at different levels of resolution under the control of a coarse-to-fine search strategy. We described three different techiques for computing the band-pass image pyramid. We also discussed the issues involved in the correlation and search processes.

We have shown how this hierarchical correlation technique both reduces the costs of correlation matching and avoids the mismatch problem that occurs in areas of high feature density. The results of our experiments indicate that it is also insensitive to noise and is able to detect at least small amounts of rotation between images.

Acknowledgements

We would like to thank Peter Burt for his foundational work and for the valuable discussions we had with him. Our thanks also go to Tom Williams and Bryant York for their detailed comments on the first draft of this paper and to Paul Nagin, Steve Epstein, and Daryl Lawton for our discussions with them.

Programming pyramid algorithms in the processing cone has been greatly facilitated by the VISIONS Image Operating System [10]. We would like to acknowledge it's designers and the many contributors within the VISIONS project at the University of Massachusetts.

References

1. Aggarwal,J.K., Davis,L.S., and Martin,W.N., Correspondence Processes in Dymanic Scene Analysis, *Proc. IEEE* 69(5):562-672, 1981.
2. Burt,P.J., Fast Filter Transforms for Image Processing, *CGIP* 16:20-51, 1981
3. Burt,P.J., Pyramid-Based Extraction of Local Image Features with Applications to Motion and Texture Analysis, Proc. SPIE Conf. on Robotics and Industrial Inspection, San Diego, 1982.
4. Burt,P.J., Fast Algorithms for Estimating Local Image Properties, Tech. Rep. IPL-TR-022, Image Processing Lab., Electrical, Comp., and Systems Eng. Dept., RPI, 1982.
5. Burt,P.J., and Adelson,E.H., The Laplacian Pyramid as a Compact Image Code, Tech. Rep. IPL-TR-025, Image Processing Lab., Electrical, Comp., and Systems Eng. Dept., RPI, 1982.
6. Burt,P.J., Yen,C., and Xu,X., Local Correlation Measures for Motion Analysis: A Comparative Study, IEEE Proc. PRIP 269-274, 1982. Also IPL-TR-024, ECSE Dept., RPI, 1982.
7. Glazer,F., Multilevel Relaxation in Low Level Computer Vision, In: *Multiresolution Image Processing and Analysis*, Rosenfeld,A. (Ed.), Springer-Verlag, 1983. Also COINS Tech. Rep. 82-30, U.Massachusetts, Amherst, 1982.
8. Hanson,A. and Riseman,E.M., Processing Cones: A Computational Structure for Image Analysis, In: *Structured Computer Vision*, Tanimoto,S. and Klinger,A. (Eds.), Academic Press, New York, 1980.
9. Lucas,B.D., and Kanade,T., An Iterative Image Registration Technique with an Application to Stereo Vision, Proc. IJCAI-7 pp:674-679, Vancouver, B.C., Canada, 1981
10. Kohler,R., and Hanson,A., The VISIONS Image Operating System, Proc. Int. Conf. Pattern Recognition, pp.71-74, Munich, 1982.
11. Kitchen,L., and Rosenfeld,A., Gray-Level Corner Detection, Tech. Rep. 887, Comp. Vision Lab., Comp. Science Center, U. Maryland, 1980.
12. Marr,D. and Poggio,T., A Computational Theory of Human Stereo Vision, *Proc. R. Soc. Lond. B*, 204:301-328, 1979.
13. Moravec,H.P., *Robot Rover Visual Navigation*, UMI Research Press, Ann Arbor, Michigan, 1981.
14. Rosenfeld,A., and Kak,A., *Digital Picture Processing*, Academic Press, New York, 1982.
15. Rosenfeld,A., and Vanderbrug,G.J., Coarse-Fine Template Matching, *IEEE Tr. SMC* 7(2):104-107, 1977.
16. Tanimoto,S., and Pavlidis,T., A Hierarchical Data Structure for Picture Processing, *CGIP* 4(2):104-119, 1975.
17. Wong,R.Y. and Hall,E.L., Sequential Hierarchical Scene Matching, *IEEE Tr. Comp.*, 27(4):359-366, 1978.

THE RELATIONSHIP BETWEEN IMAGE IRRADIANCE AND SURFACE ORIENTATION

Grahame B. Smith

Artificial Intelligence Center, SRI International
Menlo Park, California 94025

ABSTRACT

A formulation of shape from shading is presented in which surface orientation is related to image irradiance without requiring detailed knowledge of either the scene illumination or the albedo of the surface material. The case for uniformly diffuse reflection and perspective projection is discussed in detail. Experiments aimed at using the formulation to recover surface orientation are presented and the difficulty of nonlocal computation discussed. We present an algorithm for reconstructing the 3-D surface shape once surface orientations are known.

1 INTRODUCTION

When the human visual system processes a single image, e.g., Figure 1, it returns a perceived 3-D model of the world, even when that image has limited contour and texture information. This 3-D model is underdetermined by the information in the 2-D image; the visual system has used the image data and its model of visual processing to reconstruct the 3-D world. While there are many information sources within the image, shading is an important source. Facial make-up or a cartoonist's shading, is an everyday example of the way shape, as perceived by our human visual system, is manipulated by shading information.

A primary goal of computer vision is to understand this process of reconstructing the 3-D world from 2-D image data, to discover the model, or models that allow 2-D data to infer 3-D structure. The focus of this work is the recovery of the 3-D orientation of surfaces from image shading.

We present a formulation of the shape-from-shading problem, i.e., recovering 3-D surface shape from image shading, that is derived under assumptions of perspective projection, uniformly diffuse reflection,[1] and constant reflectance. This formulation differs from previous approaches to the problem in that we neither make assumptions about the surface shape [2], nor use direct knowledge of the illumination conditions and the surface albedo [3]. The cost we incur for dispensing with these restrictions is the introduction of higher-order differentials into the equations relating surface orientation and image irradiance. The benefits we gain allow us to investigate the strength of the constraint imposed by shading upon shape. Past attempts to solve the shape-from-shading problem, as well as our own efforts, have been aimed at recovering surface shape from image patches for which the reflectance (albedo) can be considered constant.

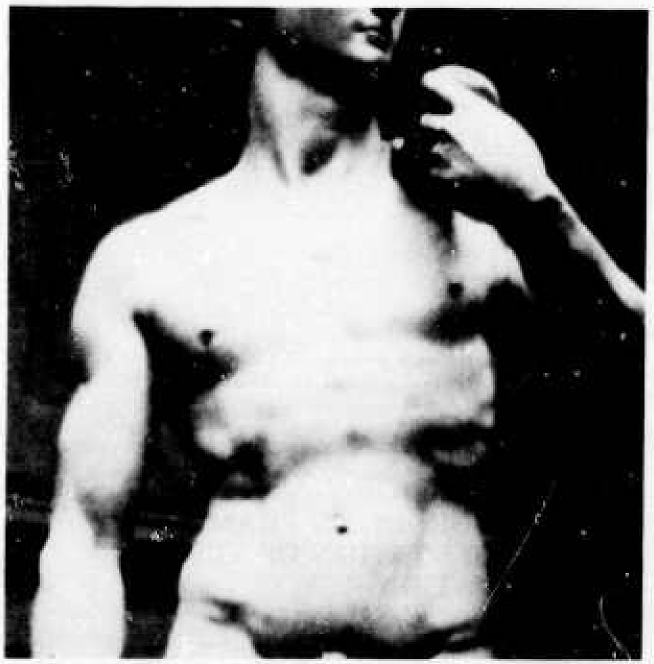


Figure 1 Shape from Shading.

Previously we examined the influence exerted by the assumption of uniformly diffuse reflection [1], and indicated that the equations relating surface orientation to image irradiance could be expected to yield useful results even in cases in which the reflection is not uniformly diffuse. In that examination we assumed orthographic rather than perspective projection. A comparison of our previous work with this paper, however, shows that the structure of the formulation is not dependent upon the projection used.

If we add additional assumptions, e.g., constraints on the surface type, we can simplify the relationship between surface orientation and image irradiance. While it is not our goal to add constraints upon surface type, the assumption that the surface is locally spherical allows the approximate surface orientation to be recovered by local computation.

The research reported herein was supported by the Defense Advanced Research Projects Agency under Contract MDA903-83-C-0027 and by the National Aeronautics and Space Administration under Contract NASA 9-16664. These contracts are monitored by the U.S. Army Engineer Topographic Laboratory and by the Texas A&M Research Foundation for the Lyndon B. Johnson Space Center.

[1] We prefer the expression *isotropic scattering* to either *uniformly diffuse reflection*, or *Lambertian reflection*, as it emphasis that scene radiance is isotropic. However, uniformly diffuse reflection, and Lambertian reflection are the terms commonly used to indicate that the scene radiance is isotropic.

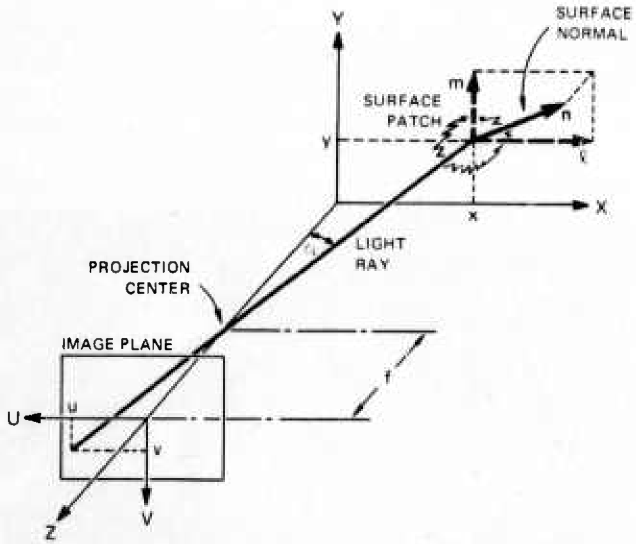


Figure 2 Coordinate Frame. X,Y,Z are the scene coordinates. U,V the image coordinates, and the image plane is located a distance f from the scene coordinate's origin - the projection center. α is the angle between the Z axis (the viewing direction) and the ray of light from the scene point (x, y, z) to the image point (u, v). l and m are the X and Y components of the surface normal $\mathbf{n}$.

2 THE COORDINATE FRAME AND REPRESENTATION OF SURFACE ORIENTATION

The coordinate system we use is depicted in Figure 2. X,Y,Z are the scene coordinates and U,V are the image coordinates. The image and scene coordinates are aligned so that X and U axes are parallel, as are the Y and V axes. The U and V axes are inverted with respect to the X and Y axes, so that positive X and Y coordinates will correspond to positive U and V coordinates. The image plane is located at a distance f from the (perspective) projection center, the origin of the scene coordinates. A ray of light from the point (x, y, z) in the scene to the image point (u, v) makes an angle α with the viewing direction (i.e., the Z axis).

There are many parameterizations of the surface orientation: we choose to use (l, m), which are the X and Y components of the unit surface normal. In Figure 2, **n** is the unit normal of the surface patch located at (x, y, z); l and m are the components of this surface normal in the X and Y directions. From our viewing position we can see at most half the surfaces in the scene (i.e., those that face the viewer). The Z component of the surface normal has the magnitude $\sqrt{1 - l^2 - m^2}$, the sign determining whether the surface is forward-facing (has a positive Z component), or backward-facing (has a negative Z component). For large off-axis angle α, we see backward-facing surfaces near the edges of objects. The two components of the surface normal, l and m, do not provide an adequate parameterization of the surface in this case. Additionally, we need to know the sign of the Z component. Here we restrict ourselves to forward-facing surfaces. This minor restriction amounts to assuming that α is not too large and that we are not adjacent to an object s edge. Consequently, in this discussion we assume that the Z component of the surface normal is positive and that l and m constitute an adequate parameterization of scene surfaces.

3 IMAGE IRRADIANCE

The image irradiance equation we use is [4]

$$I(u, v) = R(l, m) \cos^4 \alpha \quad ,$$

where $I(u, v)$ is the image irradiance as a function of the image coordinates u and v, and $R(l, m)$ is the surface radiance as a function of l and m, the components of the surface normal.[2] The term $\cos^4 \alpha$ represents the off-axis effect of perspective projection. When α is small, $\cos^4 \alpha$ is approximately unity; we then have the more familiar form of the image irradiance equation. From Figure 2 we see that

$$\cos \alpha = \frac{f}{\sqrt{u^2 + v^2 + f^2}} \quad .$$

Differentiating the image irradiance equation with respect to the image coordinates u and v, we obtain

$$I'_u = R_l l_u + R_m m_u \quad ,$$
$$I'_v = R_l l_v + R_m m_v \quad ,$$

$$I'_{uu} = R_{ll} {l_u}^2 + R_{mm} {m_u}^2 + 2R_{lm} l_u m_u + R_l l_{uu} + R_m m_{uu} \quad .$$
$$I'_{vv} = R_{ll} {l_v}^2 + R_{mm} {m_v}^2 + 2R_{lm} l_v m_v + R_l l_{vv} + R_m m_{vv} \quad ,$$
$$I'_{uv} = R_{ll} l_u l_v + R_{mm} m_u m_v + R_{lm}(l_u m_v + l_v m_u) + R_l l_{uv} + R_m m_{uv} \quad ,$$

where subscripted variables denote partial differentation with respect to the subscript(s), and

$$I'_u = (\frac{1}{\cos^4 \alpha})(I_u + \frac{4uI}{u^2 + v^2 + f^2}) \quad ,$$

$$I'_v = (\frac{1}{\cos^4 \alpha})(I_v + \frac{4vI}{u^2 + v^2 + f^2}) \quad ,$$

$$I'_{uu} = (\frac{1}{\cos^4 \alpha})(I_{uu} + \frac{8uI_u}{u^2 + v^2 + f^2} + \frac{8u^2 I}{(u^2 + v^2 + f^2)^2} + \frac{4I}{u^2 + v^2 + f^2}) \quad ,$$

$$I'_{vv} = (\frac{1}{\cos^4 \alpha})(I_{vv} + \frac{8vI_v}{u^2 + v^2 + f^2} + \frac{8v^2 I}{(u^2 + v^2 + f^2)^2} + \frac{4I}{u^2 + v^2 + f^2}) \quad .$$

$$I'_{uv} = (\frac{1}{\cos^4 \alpha})(I_{uv} + \frac{4vI_u}{u^2 + v^2 + f^2} + \frac{4uI_v}{u^2 + v^2 + f^2} + \frac{8uvI}{(u^2 + v^2 + f^2)^2}) \quad .$$

[2] Image irradiance is the light flux per unit area falling on the image, i.e., incident flux density. Scene radiance is the light flux per unit projected area per unit solid angle emitted from the scene, i.e., emitted flux density per unit solid angle.

If we are to use these expression to relate image measurements, e.g., I'_{uu}, to surface parameters l and m, then we must remove the derivatives of R.

4 UNIFORMLY DIFFUSE REFLECTION

To provide the additional constraints we need for relating surface orientation to image irradiance, we introduce constraints that relate properties of $R(l,m)$, — that is, constraints that specify the relationship between surface radiance and surface orientation. Such constraints are

$$(1-l^2)R_{ll} = (1-m^2)R_{mm} \quad ,$$
$$(R_{ll} - R_{mm})lm = (l^2 - m^2)R_{lm} \quad ,$$

where R_{ll} is the second partial derivative of R with respect to l, R_{mm} is the second partial derivative of R with respect to m, and R_{lm} is the second partial cross-derivative of R with respect to l and m.

These two partial differential equations embody the assumption of uniformly diffuse reflection. For uniformly diffuse reflection, $R(l,m)$ has the form

$$R(l,m) = al + bm + c\sqrt{1-l^2-m^2} + d \quad ,$$

where a, b, c, and d are constants, their values depending on illumination conditions and surface albedo. Note that l, m, and $\sqrt{1-l^2-m^2}$ are the components of the unit surface normal in the directions X, Y, and Z. $R(l,m)$ can be viewed as the dot product of the surface normal vector $(l, m, \sqrt{1-l^2-m^2})$ and a vector (a,b,c) denoting illumination conditions. As the value of a dot product is rotationally independent of the coordinate system, the scene radiance is independent of the viewing direction — which is the definition of uniformly diffuse reflection.

It is clearly evident that $R(l,m) = al + bm + c\sqrt{1-l^2-m^2} + d$ satisfies the pair of partial differential equations given above. In [1] we showed that $R(l,m) = al + bm + c\sqrt{1-l^2-m^2} + d$ is the solution of the pair of partial differential equations. These partial differential equations are an alternative definition of uniformly diffuse reflection.

It is worthy of note that $R(l,m) = al + bm + c\sqrt{1-l^2-m^2} + d$ includes radiance functions for multiple and extended illumination sources, including that for a hemispherical uniform source such as the sky. Of course, at a self-shadow edge R is not differentiable, so that the surfaces on each side of the self-shadow boundary have to be treated separately. The assumption of uniformly diffuse reflection restricts the class of material surfaces being considered, not the illumination conditions.

From the constraints for uniformly diffuse reflection, we derive the relationships

$$R_{ll} = \frac{1-m^2}{lm} R_{lm} \quad ,$$
$$R_{mm} = \frac{1-l^2}{lm} R_{lm} \quad .$$

Substituting these relationships for R_{ll} and R_{mm} in the expressions for I'_{uu}, I'_{vv}, and I'_{uv}, we obtain

$$[l_u{}^2(\frac{1-m^2}{lm}) + m_u{}^2(\frac{1-l^2}{lm}) + 2l_u m_u]R_{lm} =$$
$$I'_{uu} - R_l l_{uu} - R_m m_{uu} \quad ,$$
$$[l_v{}^2(\frac{1-m^2}{lm}) + m_v{}^2(\frac{1-l^2}{lm}) + 2l_v m_v]R_{lm} =$$
$$I'_{vv} - R_l l_{vv} - R_m m_{vv} \quad ,$$
$$[l_u l_v(\frac{1-m^2}{lm}) + m_u m_v(\frac{1-l^2}{lm}) + l_u m_v + l_v m_u]R_{lm} =$$
$$I'_{uv} - R_l l_{uv} - R_m m_{uv} \quad .$$

By removing R_{lm} and substituting the expressions for R_l and R_m, defined by the expressions for I'_u and I'_v, we produce two partial differential equations relating surface orientation to image irradiance:

$$\alpha\theta l_{uu} + \beta\theta m_{uu} - \alpha\gamma l_{uv} - \beta\gamma m_{uv} = \chi\theta I'_{uu} - \chi\gamma I'_{uv} \quad ,$$
$$\alpha\theta l_{vv} + \beta\theta m_{vv} - \alpha\delta l_{uv} - \beta\delta m_{uv} = \chi\theta I'_{vv} - \chi\delta I'_{uv} \quad ,$$

where

$$\alpha = I'_u m_v - I'_v m_u \quad ,$$
$$\beta = I'_v l_u - I'_u l_v \quad ,$$
$$\gamma = l_u{}^2(1-m^2) + m_u{}^2(1-l^2) + 2l_u m_u lm \quad ,$$
$$\delta = l_v{}^2(1-m^2) + m_v{}^2(1-l^2) + 2l_v m_v lm \quad ,$$
$$\theta = l_u l_v(1-m^2) + m_u m_v(1-l^2) + (l_u m_v + l_v m_u)lm \quad ,$$
$$\chi = l_u m_v - l_v m_u \quad .$$

These equations relate surface orientation to image irradiance by parameter-free expressions. We make no assumptions about surface shape, nor do we need to know the parameters specifying illuminant direction, illuminant strength, and surface albedo. Our assumptions are about the properties of reflection in the world; these alone are sufficient to relate surface orientation to image irradiance. The above equations have been derived for the case of perspective projection; for orthographic projection, the primed ($'$) quantities are replaced by their unprimed counterparts, e.g., I'_u is replaced by I_u. The form of the equations is not a function of the projection used.

5 RECOVERY OF SURFACE ORIENTATION

It is difficult to solve the equations relating surface orientation to image irradiance, and thus to recover surface shape from observed image irradiance. We have used numerous integration schemes that characterize two distinct approaches. The two differential equations can be directly integrated in a step-by-step manner or, given some initial solution, a relaxation procedure may be employed. The difficulties that arise are twofold: numerical errors and multiple solutions.

Solutions of the equation $\chi = 0$ (the developable surfaces, e.g., a cylinder) are also solutions of the equations relating surface orientation to image irradiance. If the image intensities were known in analytic form, the analytic approach to solving the equations could then employ boundary conditions to select the appropriate solution. However, since the analytic form for the image intensities is unknown, numerical procedures must be employed The use of such procedures to directly integrate the equations inevitably introduces small errors. Such errors 'mix in' multiple solutions even when those solutions are incompatible with the boundary conditions. Instability of the numerical scheme seems responsible for the fact that such errors eventually dominate the recovered solution. A scheme that is representative of our various trials at direct integration is outlined.

We transform our equations into finite-difference equations by using a three-point formula for the differentials of l and m. If $l(i,j)$ and $m(i,j)$ are the values of l and m at the (i,j)th pixel in the image, then at this pixel we use the finite-difference formulas,

$$l_u = \frac{l(i+1,j) - l(i-1,j)}{2} \quad ,$$

$$l_{uu} = l(i+1,j) + l(i-1,j) - 2l(i,j) \quad ,$$

$$l_{uv} = \frac{l(i+1,j+1) + l(i-1,j-1)}{4} - \frac{l(i+1,j-1) + l(i-1,j+1)}{4} \quad ,$$

and similar formulas for the other differentials. If we consider the 3 x 3 image patch centered on the (i,j)th pixel,

	j-1	j	j+1
i+1	O	O	&
i	O	O	O
i-1	O	O	O

we could hope that the two finite difference equations, relating the eighteen values of l and m on the patch, could be solved explicitly for $l(i+1,j+1)$ and $m(i+1,j+1)$, (the (&) cell). Such a solution would allow l and m at the (&) cell to be calculated from the l's and m's at the (o) cells. Starting at some boundary at which we know l and m at the (o) cells, we can move along the image's row and then along the successive rows, calculating l and m at the (&) cell. However, examination of the surface-orientation-to-image-irradiance equations shows that we cannot solve these equations explicitly for l_{uv} and m_{uv} and that, consequently, we cannot obtain finite-difference equations that are explicit in the l and m of the (&) cell.

We avoid this difficulty by combining the two surface-orientation-to-image-irradiance equations into one and using surface continuity to provide the additional equation. Removing l_{uv} and m_{uv} from the differential equations, we have

$$\alpha(\delta l_{uu} - \gamma l_{vv}) + \beta(\delta m_{uu} - \gamma m_{vv}) = \chi(\delta I'_{uu} - \gamma I'_{vv}) \quad .$$

Surface continuity requires that $\frac{\partial^2 z}{\partial x \partial y} = \frac{\partial^2 z}{\partial y \partial x}$, from which it follows that

$$l_y(1-m^2) + m_y lm = m_x(1-l^2) + l_x lm \quad .$$

Provided that u and v are small compared with z (e.g., in the eye or in a standard-format camera), then

$$l_v(1-m^2) + m_v lm = m_u(1-l^2) + l_u lm \quad .$$

These two equations, which do not involve l_{uv} or m_{uv}, form a basis for finite difference equations that calculate l and m at the (-) cell from values of l and m at (+) cells.

	–	
+	+	+
	+	

The results obtained with the above integration scheme, together with many variations of it, are poor. Accurate values for l and m are obtained only within approximately five to ten rows of the known boundary. This is the case for noise-free image data. These results can be understood by examination of the finite-difference equations. The explicit expressions for l and m at the (-) cell are functions of the differences of l and m at the (+) cells. Such schemes are usually numerically unstable, making step-by-step integration impossible. While the failure to find a stable numerical scheme does not imply that one does not exist, our difficulty highlights the problem of finding numerical schemes, based on differential models, to propagate information from known boundaries. (One wonders whether nature experienced the same difficulties when designing the human vision system.)

Although the alternative to direct integration, a relaxation procedure to solve the equations, seems to offer relief from the numerical instability of direct integration, it nevertheless poses its own problems. The approach we used parallels the one in [3] for solving the image irradiance equation when the surface albedo and illumination conditions are known. For each image pixel we form three error terms: the residuals associated with the two surface-orientation-to-image-irradiance equations, and with the one surface continutiy equation. Minimizing the sum of the errors over the whole image with respect to l and m at each pixel produces an updating rule for l and m at each pixel. Given an initial solution, i.e., assignment of values for l and m at each pixel, a relaxtion scheme, like the one described, is useful only if it converges. While the constraint imposed by the underlying model is most important in ensuring convergence, the importance of a good initial solution for a relaxation method cannot be overemphasized. Simplifying the two partial differential equations (by using additional assumptions) provides a method for obtaining an good initial solution.

The spherical approximation assumes that we are viewing a spherical surface. This implies $l_y = 0, m_x = 0$, and $l_x = m_y$, — namely, constant curvature that is independent of direction. Provided that u and v are small compared with z, then $l_v = 0, m_u = 0$ and $l_u = m_v$. For this case, the partial differential equations become relationships between image irradiance and its derivatives, on the one hand, and the components of the surface normal, on the other:

$$\frac{1-m^2}{lm} = \frac{I'_{uu}}{I'_{uv}} \quad ,$$
$$\frac{1-l^2}{lm} = \frac{I'_{vv}}{I'_{uv}} \quad .$$

The spherical-approximation results for perspective projection are similar to those Pentland was able to obtain [2] for orthographic projection through local analysis of the surface. Besides providing a mechanism for obtaining an initial solution for a relaxation-style algorithm, they allow surface orientation to be estimated by purely local computation. Such an estimate will be exact when the surface is locally spherical.

The results of our experiments with relaxation procedures are easily summarized: the relaxation procedures were not convergent. While such nonconvergence is hardly unusual, the reasons for failure, however, are instructive. The residuals associated with both the surface-orientation-to-image-irradiance equations, and the surface continuity equations remain small during the relaxation, even when the solution is starting to diverge. Of course the residuals are not as small as they are when on the verge of solution, but they are small enough to make one believe that a solution has been obtained, particularly when the image is not noise-free. Apparently the equations are insensitive to particular values of l and m, being more concerned with the values of l_u,l_v,m_u, and m_v. As with direct integration, relaxation models need boundary conditions to select a particular solution. We used various boundary conditions in our relaxation experiments, but it is difficult to believe that a model, apparently insensitive to surface orientations, could be overly influenced by the surface orientations at a boundary.

Our two approaches, direct integration and relaxation, have not yielded a computational solution to the problem of recovering surface orientation from shading. The attractiveness of local computation is clear; it has neither numerical instability nor divergent behavior, but the cost it imposes is that assumptions must be made about surface shape. A compromise between some local computation and some information propagation may offer an approach that is not overly restrictive in its assumptions about surface shape. However, the question needs to be considered: Is the model underconstrained? Is shape recovery dependent on information other than shading? What other information (that is obtainable from the image), is necessary to enable the construction of effective shape-recovery algorithms?

6 RECONSTRUCTION OF THE SURFACE SHAPE

Surface orientation is not the same as surface shape. However, once we have obtained the surface orientation as a function of image coordinates, i.e., $l(u,v)$ and $m(u,v)$, we can use these to reconstruct the surface shape in the scene coordinates X,Y,Z. We derive a suitable formula.

Suppose we know the depth z_0 at scene coordinates (x_0, y_0, z_0), corresponding to (u_0, v_0) in the image. For the point $(x_0+\Delta x, y_0+\Delta y)$ we use the approximation

$$z(x_0+\Delta x, y_0+\Delta y) = z(x_0,y_0) + \Delta x \left.\frac{\partial z}{\partial x}\right|_{x_0,y_0} + \Delta y \left.\frac{\partial z}{\partial y}\right|_{x_0,y_0} \quad .$$

Similarly,

$$z(x_1-\Delta x, y_1-\Delta y) = z(x_1,y_1) - \Delta x \left.\frac{\partial z}{\partial x}\right|_{x_1,y_1} - \Delta y \left.\frac{\partial z}{\partial y}\right|_{x_1,y_1} \quad .$$

If $x_1 = x_0 + \Delta x$ and $y_1 = y_0 + \Delta y$, then

$$z(x_1,y_1) = z(x_0,y_0) + \frac{x_1-x_0}{2}\left(\left.\frac{\partial z}{\partial x}\right|_{x_0,y_0} + \left.\frac{\partial z}{\partial x}\right|_{x_1,y_1} \right) + \frac{y_1-y_0}{2}\left(\left.\frac{\partial z}{\partial y}\right|_{x_0,y_0} + \left.\frac{\partial z}{\partial y}\right|_{x_1,y_1} \right) \quad .$$

Using the perspective transformation $u = -f\frac{x}{z}$ and $v = -f\frac{y}{z}$ to remove x and y, we obtain

$$z(u_1,v_1) = z(u_0,v_0)\mathbf{x} \frac{2f + u_0\left(\left.\frac{\partial z}{\partial x}\right|_{u_0,v_0} + \left.\frac{\partial z}{\partial x}\right|_{u_1,v_1} \right) + v_0\left(\left.\frac{\partial z}{\partial y}\right|_{u_0,v_0} + \left.\frac{\partial z}{\partial y}\right|_{u_1,v_1} \right)}{2f + u_1\left(\left.\frac{\partial z}{\partial x}\right|_{u_0,v_0} + \left.\frac{\partial z}{\partial x}\right|_{u_1,v_1} \right) + v_1\left(\left.\frac{\partial z}{\partial y}\right|_{u_0,v_0} + \left.\frac{\partial z}{\partial y}\right|_{u_1,v_1} \right)} .$$

As $\frac{\partial z}{\partial x} = \frac{-l}{\sqrt{1-l^2-m^2}}$ and $\frac{\partial z}{\partial y} = \frac{-m}{\sqrt{1-l^2-m^2}}$, we have the means of reconstructing the surface in scene coordinates from the values of surface orientation in image coordinates.

7 CONCLUSION

In this formulation of the shape-from-shading task, we have eliminated the need to know the explicit form of the scene radiance function by introducing higher-order derivatives into our model. This model is applicable to natural scenery without any additional assumptions about illumination conditions or the albedo of the surface material. However, without a computational scheme to reconstruct surface shape from image irradiance we may wonder if we have surrendered too much. The difficulties of finding a computational scheme must induce one to ask whether the model is underconstrained. Have we applied too few restrictions, thereby making shape recovery impossible? Notwithstanding the general concern about underconstraint of the model, the numerical difficulties encountered makes local computation of scene parameters attractive. Information propagation methods must always cope with the problem of accumulated errors. In our model, however, to achieve local computation we must make assumptions with regard to surface shape. What other information, besides shading, do we need to know if we are to recover surface shape? Can we find moderate restrictions that allow mostly local computation of the surface shape parameters? We are actively engaged in the pursuit of such procedures.

REFERENCES

1. Smith, G.B., From image irradiance to surface orientation, (submitted for publication). Available as Technical Note 273, Artificial Intelligence Center, SRI International (1982).
2. Pentland, A.P., The visual inference of shape: computation from local features, Ph.D. Thesis, Department of Psychology, Massachusetts Institute of Technology (1982).
3. Ikeuchi, K. and Horn, B.K.P., Numerical shape from shading and occluding boundaries, *Artificial Intelligence* **17** (1981) 141-184.
4. Horn, B.K.P. and Sjoberg R.W., Calculating the reflectance map, A.I. Memo 498, Artificial Intelligence Laboratory, Massachusetts Institute of Technology (1978).

ENVIRONMENTAL RELATIONS IN IMAGE UNDERSTANDING: THE FORCE OF GRAVITY

John R. Kender
Department of Computer Science, Columbia University
New York, NY 10027

Abstract

It is detailed

In this paper we show how assumptions and information concerning the external world properties of "horizontal" and "vertical" can aid in the analysis of images, even at the very lowest levels of processing. First, we review the pervasiveness of the force of gravity, and its influence on most natural image understanding systems. Next, we derive several fundamental mathematical results relating phenomena in both the gradient space and the image space to the external world attributes of horizontal and vertical. We then show how these results interrelate three imaging phenomena: the surfaces in the image, the external sensor parameters, and the environmental labels. We detail how, in general, specific information regarding any two of these phenomena can be used to quantitatively derive the third; occasionally one can do even better. Algorithms for such quantitative derivations are presented, including two based on the Hough transform. We further show how certain environmental perpendicularities can be exploited very efficiently, and even elegantly: ordinarily complex math simplifies to the extent that environmental distances can be directly read off the image. The power of such environmental labels is then demonstrated by an analysis of the source of ambiguity in a simple illusion-like image configuration. The paper concludes with an analysis of the class of heuristics that have been invoked throughout. They are seen to be instantiations of the shape-from-texture meta-heuristics that "near implies preferred" and "preferred implies simple".*

The author

1 Introduction

Many image environments are immersed in a force that strongly orients objects in a preferred way. The effects of this force are often so pervasive that environments which do not respond to it appear (and are often called) artificial. The very term "natural scene", vague though it may be, does at least seem to imply an image with just such a definite environmental orientation. Researchers would no sooner attempt to fully analyze such an image upside-down than they would if its colors had been permuted.

It is not difficult to be convinced of the influence that the presence of gravity has on the design of image understanding algorithms, especially in higher level processing. Often it is so strong that it deeply permeates the entire system as an implicit assumption. The assumption is made with good reason: higher level processing can be more efficient. Matching to models, for example, can start with both the detected object and the modeled object mutually aligned in the preferred (that is, the most probable) orientation.

However, we show in this paper that assuming the presence of gravity can aid the lower levels of image processing as well. This is a bit surprising, since many low-level routines do work--and ought to work--just as well with images inverted (or colors scrambled). Nevertheless, certain heuristics regarding the exploitation of "horizontal", "vertical", and other gravity-influenced concepts can make low-level shape recovery more efficient as well.

These heuristic assumptions, coupled with some fundamental mathematical results, can suggest methods and algorithms on the same level as other "shape from" methods, such as shape from shading or skewed symmetry [Horn 77; Woodham 78; Kender 80a; Ikeuchi 80]. The heuristics themselves usually are based on the assumption of some preference: here, the preference for horizontal or vertical surfaces or lines. Thus, they can be seen as further members of the family of preference-based algorithms linked together by their derivation and use in a common methodological paradigm, called shape from texture [Kender 80b].

2 The Pervasiveness of Gravity

The presence of gravity introduces and maintains in "natural" environments a decided anisotropy. Its lines of force are parallel to each other in one specific, unchanging orientation. This orientation induces, usually by means of general energy minimization arguments, configurations that are themselves parallel: natural (as well as artificial) growth is often aligned with the field. Thus trees as well as buildings often have parallel sides, and are parallel to each other. Further, the "ground plane" is often actually planar, also in a minimizational reaction to the force: whether it is truly the ground, or artificially made so, as in a floor. The combination of these growth parallelism and ground planes further induce perpendicularities, again both natural and artificial. The junction of trees or animal legs to forest floor (or to their shadows), or the junctions of walls to ceilings (or object legs to floors), all occur in a limited class of orientations.

Other examples of gravity's explicit and implicit involvement with image understanding can easily be given. In some domains, of course, it has no influence at all: for example, blood cell analysis. However, in most "natural" domains--or, equivalently, most "robotic" domains--its pervasiveness appears to be so extensive that a "natural" scene might very well be *defined* as one in which considerations of gravitationally induced orientations are non-negligible. In other words, a scene is a natural scene to the degree that it would be difficult to understand rotated or upside-down. (Thus, images of office interiors are about as natural as handwriting samples; both are more natural than most aerial photography; high magnification scanning electron micrographs are least natural of all.)

*This research was sponsored in part by the Defense Advanced Research Projects Agency under contract N00039-82-C-0427.

3 Gradient Space Relations

Perhaps the first basic relationship that deals with the environmental labels "horizontal" and "vertical" are the terms used to define the degrees of freedom of the sensor itself. The sensor orientation terms "pan", "tilt", and "roll" imply a gravity-dependent coordinate system, and, in fact, are defined in environmental terms. Pan is sensor rotation in the horizontal plane; tilt is rotation in the vertical plane passing through the central visual ray. Roll is defined as rotation in the image plane, and its effect is therefore dependent on tilt and pan; in the absence of roll, the image of an environmentally vertical plane that passes through the central visual ray is a retinally vertical line.

A second basic relationship is that, in terms of its use in computer vision, "horizontal" is simply a label for a unique, preferred surface orientation. In terms of the gradient space [Shafer 83a], it is a single labeled orientation point with coordinates $(p,q) = (p_h,q_h)$. Assuming that there is no roll in the sensor--that is, the y-axis of the image is the projection of an environmentally vertical plane--then this point simplifies to $(p,q) = (0,q_h)$.

This relationship is schematically depicted in Figure 1. The value of q_h is easily determinable: assuming the sensor is at a unit's distance from the ground plane and has no roll, then the central visual ray intersects the ground at q_h. Note that this value can also be obtained by a simple gravity sensor. The y-axis now lies in an environmentally vertical plane as in Figure 2; further, due to the rotational coupling of the gradient space to the image space [Kender 80b], the horizontal orientation has no p component.

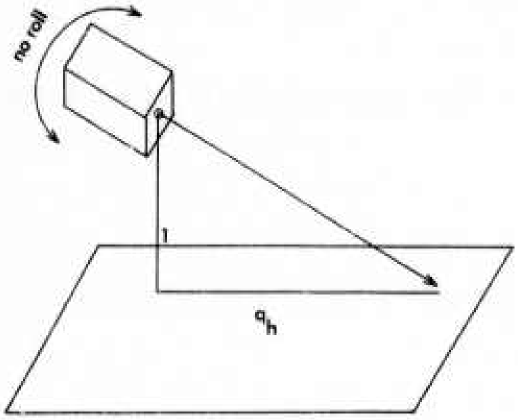


Figure 1: Basic relations: sensor configuration.

It is not hard to show that every vertical surface must map into a gradient space point with coordinates $(p,q) = (p,-1/q_h)$. This fact follows from the general rule that the gradients of surfaces perpendicular to a given gradient (p_h,q_h) must satisfy the relation $pp_h+qq_h = -1$; every vertical surface is perpendicular to the horizontal. Thus, the one-dimensional family of vertical surfaces maps into the one-dimensional locus $q = -1/q_h$ as in Figure 3 [Mackworth 73]. As a special case, if there is neither sensor roll nor tilt then the gradient space representation for the horizontal surface is infinitely far along the positive q axis, and the line of verticals becomes the p axis.

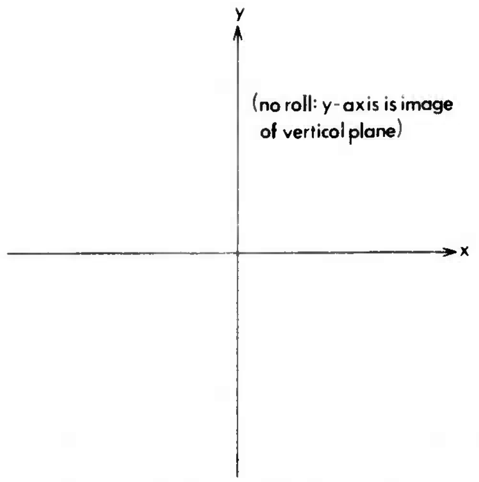


Figure 2: Basic relations: Corresponding image space.

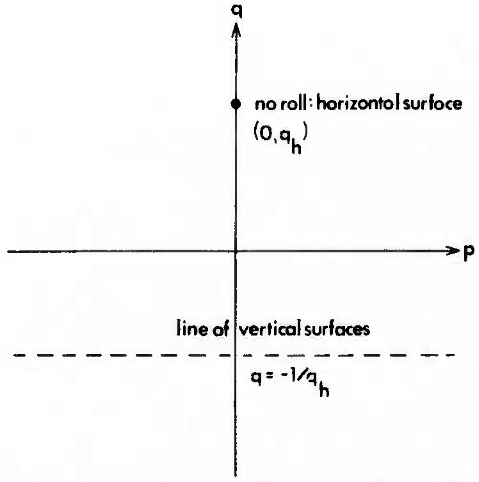


Figure 3: Basic relations: Corresponding gradients.

More generally, similar basic relationships hold even if there is a roll component. It is not hard to show that if there is information available about the sensor's tilt and roll, then the gradient space can be environmentally labeled by invoking the rotational coupling of the image space to the gradient space. That is, if tilt is given as above by the angle whose tangent is q_h, and the roll component is given as t with respect to the unrolled sensor position, then the point in the gradient space corresponding to the horizontal is given by $(0,q_h)$ similarly rotated through t. The line of verticals rotates likewise.

It is important to note that the above relations hold independently of any considerations of imaging projection. They are true for both orthography and perspective; they are true, in fact, even with no image at all. They describe the relations of the gradient space to environmental preference labels only. (Alternatively, one can use the analogous relations that prevail when surface orientations are recorded on a Gaussian sphere map [Horn 82]). Further, they can be used in either direction: given sensor information, the gradient space can be labeled, and vice versa.

4 Image Space Relations

At this point, we have not yet used any image information. In fact, in as much as surfaces exist in three-space, they cannot appear directly in an image at all. However, it is interesting to note that the same environmental labels of horizontal and vertical apply to lines as well, and to both lines in three-space and lines on the retina. Somewhat paradoxically, though, the size of the class of environmentally horizontal lines is one dimension greater than that of environmentally vertical ones; this is the reverse of the case with surfaces.

Environmental labels, environmental line segments, and the sensor parameters are related in several ways. To demonstrate them, consider first the case of perspective imaging where the sensor has no roll component. Scale the image plane in units of focal length; this will simplify the mathematics [Kender 80b]. Now image a scene consisting of vertical lines emerging from a horizontal plane: rather like a vast, stylized forest. The result is shown schematically in Figure 4.

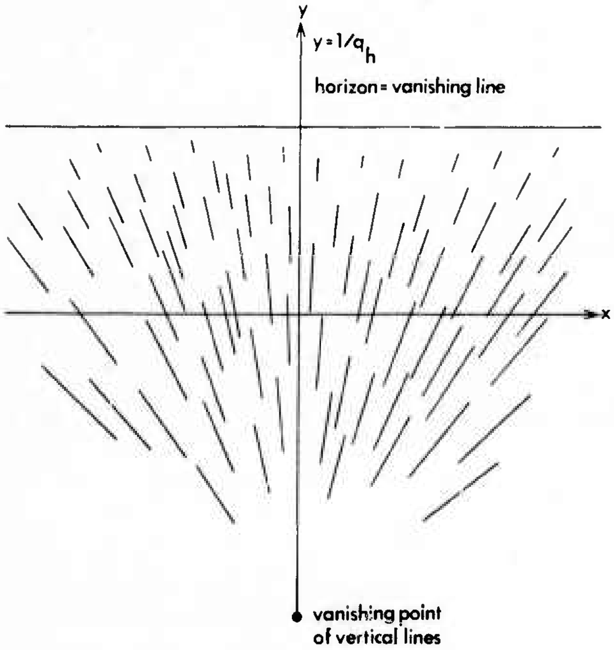


Figure 4: Vertical lines on a horizontal surface.

Because the class of environmentally horizontal lines is so large, they retain no distinguishing retinal features. That is, any line in the image can be the image of an environmentally horizontal line. About the only exploitable horizontal property is the horizon itself. This line, the limit of the projection of the horizontal plane, is a retinally horizontal. It has the equation $y = 1/q_h$. This follows from the basic relationship concerning vanishing lines: the plane with gradient (p,q) has vanishing line $px+qy = 1$, here $(p,q) = (0,q_h)$.

More interesting is the behavior of the environmental verticals. They form a more restricted class, and their images are more constrained. In particular, any environmentally vertical line must image into a retinal line that passes through the point $(x,y) = (0,-q_h)$. This follows as a special case of the analysis of vanishing points [Kender 79]. As the sensor's tilt increases so that its central visual ray approaches the vertical (i.e. as q_h approaches 0), this vanishing point of verticals approaches the image origin; simultaneously the horizon moves off in the positive y direction.

If the forest scene is imaged by an orthographic sensor, very little environmental information remains in the image. Nothing at all remains of the horizon. All environmentally vertical lines are imaged as retinally vertical lines; they have no finite vanishing point. Therefore, under orthography there are no image cues to sensor tilt.

As with the gradient space relations, these image relations hold analogously under sensor roll. If the sensor is orthographic, then the parallel family of image verticals roll proportionately. If the sensor uses perspective, then the horizon and the vanishing point of verticals also roll proportionately and their expected locations are easy to compute, given tilt. The close relation between tilt and roll and the generated horizon is well known; it is exploited in the artificial horizon instruments of airplane cockpits.

Note that unlike the gradient space relations, however, these relations are not automatically reversible. That is, a given sensor configuration predicts definite image phenomena, but a given image phenomenon does not necessarily imply a sensor configuration. However, if the phenomenon can be environmentally labeled accurately (i.e. "horizon", "vertical vanishing point") then the implications about the sensor are correct. In general, though, this labeling must be done heuristically, as described below.

5 Using the Gradient Space Relations

The relationships described above can be exploited in many ways. For example, given the sensor configuration, one can recover an environmentally labeled gradient space map as in Figure 3. If the sensor configuration is uncertain, the gradient space map (more simply, the gradient of a properly labeled horizontal surface) can be used to help calibrate tilt and roll.

(It should be noted that the pan parameter can not recovered. In a sense, pan is "gravity invariant". That is, there is no information in an environmentally labeled gradient space map that would indicate pan. Pan does not even have any common environmental names. Perhaps the closest terms would be those used to describe compass directions: "north-by-northwest", etc. However, the magnetic force on which they are based seem to have negligible environmental influence; few natural systems appear capable of detecting it. The natural world does not seem to have a strong left-right preference. For example, although it is nearly impossible to find a newspaper photograph that has been printed upside-down, it is not unusual to find one that has been "flopped" left-for-right. Nevertheless, there may be artificial environments in which it would be useful to augment a mobile robot's gravity sensor with a compass.)

Additional uses of these relations include the following. If the sensor parameters are known, then the determination that a given surface is horizontal uniquely specifies it gradient. The determination that it is vertical creates a linear constraint in the gradient space on which its gradient must lie. This constraint can be used with any other gradient space constraints: for example, those obtained by shape from shading, skewed symmetry, or shape from texture.

If the sensor parameters are unknown, then a determination that two non-parallel surfaces are vertical yields tilt and roll: their gradients generate the line of verticals in the gradient space. The determination of a single surface being vertical constrains tilt and roll to one degree of freedom; horizontal surfaces must be perpendicular to it.

5.1 A Hough-like Algorithm

Suppose we pose the more difficult problem in which there is *neither* a labeling nor sensor information. Nevertheless, both can still be (heuristically) recovered. Consider the additional assumption that all (or most) surfaces are either horizontal or vertical--an assumption often supportable in man-made environments. Then the gradient space representation (or the Gaussian map) of the surfaces in the scene can be analyzed for the presence of the characteristic point-of-horizontal/line-of-verticals configuration. This need not be an actual search for the line of verticals, although there may be some environments in which this is an efficient thing to do. Instead, it can be achieved using a type of Hough accumulator approach.

In broadest outline, this method has all existing surfaces vote for candidate horizontal surfaces. Once voting is done, the surface with the most votes is then presumed to be horizontal. Sensor tilt and roll, and the line of verticals are easily determined.

Voting is prescribed in the following way. Since a given surface is likely to be either horizontal or vertical, it votes once for itself since it may itself be horizontal. However, since it may also be vertical, it votes once for all surfaces perpendicular to it: in this case, at least one of these surfaces must be the horizontal. Graphically, this is displayed in Figure 5. A vote for the self is shown by a circle about the point; a vote for all perpendiculars is indicated by a dashed line. In the example, only one surface has received four votes; it is assumed to be horizontal.

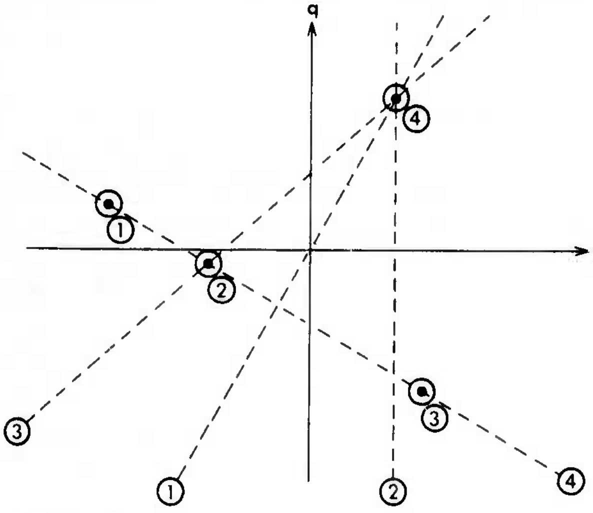


Figure 5: Hough scheme for finding ground planes.

5.2 A Critique of the Algorithm

This method is not without its problems, but it does have a virtue or two. The problems are manifest. Most critically, any such weighting scheme is heavily dependent on the given gradient space map, which in turn is affected by the environment and by the sensor position. Thus, if there is only one surface present, or even if there are two mutually perpendicular ones, there are no grounds by which to label anything horizontal. If there are two non-perpendicular surfaces present, the method considers them both vertical to a common horizontal (which does not appear in the gradient space.) If there are multiple surfaces, the voting is affected by the way in which the multiple surfaces have been recorded in the gradient space map: perhaps this map itself has been weighted. Lastly, the method is subject to the time and space problems that all Hough methods are plagued with: the space must be carefully quantized (a problem which is less severe on the Gaussian sphere), the line of votes must be calculated, votes must be distributed among accumulators proportionately, etc.

But the method does have some justifications. In particular, like most Hough transforms it can be made heuristically more efficient, and it is likely to be robust with respect to noise--which in this case are surfaces which are neither horizontal or vertical. Further, it works with surfaces that are curved verticals: building support columns, say, or drapery. In these cases, the gradient space map of the vertical surfaces is diffused along a line. Nevertheless the voting proceeds accurately, with each small quantum of the diffusion adding its small votes for its own perpendiculars. Perhaps most interesting is the result that the horizontal can be found even if there is no direct evidence for it in the gradient space: the ground can be "seen" even though it is "not there", as in Figure 6. This occurs when many environmentally vertical surfaces all vote for their perpendiculars; the one perpendicular they have in common must be horizontal, whether it is present in the gradient space map or not.

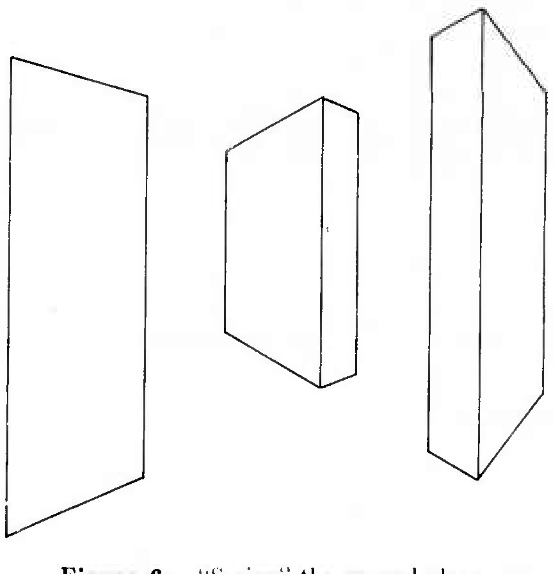


Figure 6: "Seeing" the ground plane.

6 Using the Image Space Relations

The relations concerning image configurations can also be exploited in many ways. The simplest case is when all sensor information is known. One immediate result is that locations of both the horizon and the vanishing point of verticals are then also known, whether or not any phenomena suggesting them actually appear in the image. (If either *are* suggested by an image configuration, then that configuration can be assumed to be the proper one: its position can further calibrate sensor tilt and roll.)

Again, the more interesting algorithms occur when the sensor information is unknown, uncertain, or known only partially. Under perspective, if the focal point of the retina and the focal length of the sensor are known, then sensor tilt and roll can be found immediately from a line that has been correctly labeled as the horizon. The same is true if a pair of non-colinear lines are environmentally labeled as vertical: the intersection of their extensions give the vanishing point of verticals, and hence tilt and roll. (As with the gradient space, no image information provides pan.) If there is only one such vertical, then the tilt and roll are constrained to one degree of freedom; since the vanishing point of verticals can occur anywhere on this line, this constraint corresponds to the "vanishing gradient" of the line as defined in [Shafer 83a].

Still under perspective, if focal point, focal length, and roll are known, then only one environmentally vertical line suffices to obtain tilt. In this case, the line that would be the image of a vertical plane through the focal point can be hallucinated; its intersection with the given vertical gives the vanishing point, which gives tilt.

Under orthography, neither the focal point nor focal length are required; they are "everywhere" and "infinity", respectively. But only roll is recoverable since the images of environmentally vertical lines no longer converge. However, roll can now be determined from a single image line that has been properly labeled as an environmental vertical.

Two questions remain: how are focal point, focal length, and roll obtained if they are unknown, and how are lines environmentally labeled?

Although complete sensor information is often available, occasionally--as in the case of an isolated photograph or a freely positioned sensor--some of it is not. Given that sense can usually be made of such environmentally detached images, it must be true that heuristics can be used to help quantify the missing parameters. There are many such means available, since the problem can be addressed at all processing levels of image understanding. For example, "ground truth" can help calibrate a sensor [Fischler 81]. If one is dealing with man-made environments, one can use assumptions of multiple in-plane parallelisms and mutual inter-plane perpendicularities to obtain vanishing points; these constrain sensor location, sensor attitude, focal point, and focal length [Kender 80b].

However, even at the lowest level of algorithms, fairly simple, purely environmental heuristics are possible. In the case of an isolated photograph, the focal point can be assumed to be the center of the photograph, and the focal length can be assumed to be a fixed ratio of the actual photograph dimensions. That is, the photograph can be assumed not to have been cropped.

The more pressing problem is with that of the free-floating sensor: the heuristic environmental labeling of lines for the determination of roll. Of course, given an uncropped image, one can always assume that was *no* roll, and that the image of the environmentally vertical plane through the focal point would appear as the image's vertical midline.

But one can perhaps do a bit better by first assuming that all retinally near-vertical lines, for suitable definitions of "near", are the images of environmentally vertical ones. Under perspective, a Hough-like scheme can then be used to help refine this heuristic labeling. Extend (and weight) all such lines, and take their most common intersection as the vertical vanishing point. Lines that do not pass through it loose the label. Tilt and roll come free. Under orthography, the method degenerates to one-dimensional histogramming: since the images of true environmental verticals must be parallel, one only labels those lines that have the modal (the roll) orientation. These methods share all the usual properties of a Hough transform, good and bad. (The analogous methods for environmentally horizontal lines appear to be much weaker, given their unconstrained behavior in the image.)

A special case of heuristic labeling of vertical lines is used in the Phoenix system [Herman 82]. It works on aerial views of buildings in Washington, D.C., taken with a sensor aligned directly downward. The vanishing point for verticals is therefore the image origin. Any line whose extension passes through the image origin is heuristically labeled a building's vertical.

7 Using Linear Perpendicularities

The force of gravity also induces in the environment several types of perpendicularities. We have already described several algorithms that exploit the perpendicularities that are created upon horizontal surfaces by vertical surfaces or lines. We now show several ways in which to exploit the perpendicularity created upon horizontal *lines* by vertical lines. In the discussion that follows, we assume all sensor parameters are known. To simply the presentation, we further assume that there is no sensor roll, although nothing to follow depends on that fact.

In general, these algorithms are based on the observation that an environmentally horizontal line meets an environmentally vertical line at a right angle, and creates a vertical plane. Additionally, if the sensor parameters are known, the images of environmentally vertical lines can be hallucinated in abundance: they must only pass through the vanishing point of verticals (which is an ideal point in the case of orthography). Thus, all that is needed to define a vertical surface is the actual image of an environmentally horizontal line.

The gradient of this hallucinated surface can be quantified. The knowledge that the plane is vertical already restricts its gradient to the line of vertical surfaces in the gradient space. But the fact that the environmental angle is a right angle itself generates another one-dimensional constraint in the gradient space [Kender 80b]. These two constraints can be intersected, usually resulting in a small, discrete number of gradients for the local vertical surface. Depending on the imaging geometry and the properties of the given horizontal, this gradient can be specified uniquely. In short, a line labeled as environmentally horizontal generates a well-specified vertical surface.

The second constraint, generated under the information that the image angle it is environmentally right, is complex: it is a fourth degree curve in gradient space parameters p and q. However, under orthography, or under perspective if the vertex of the angle lies on the focal point (local orthography), it simplifies to the Kanade hyperbola in p and q [Kanade 79]. A further simplification occurs since one side of the angle is environmentally vertical: the angle can be drawn in both cases as in Figure 7. The vertex is at the image origin and the environmentally vertical side is aligned with the image of the vertical plane passing through the focal point. (With no roll, this image is the y-axis). One figure suffices for both the orthographic and special perspective cases because under orthography any image can be translated to the origin without affecting the gradient space.

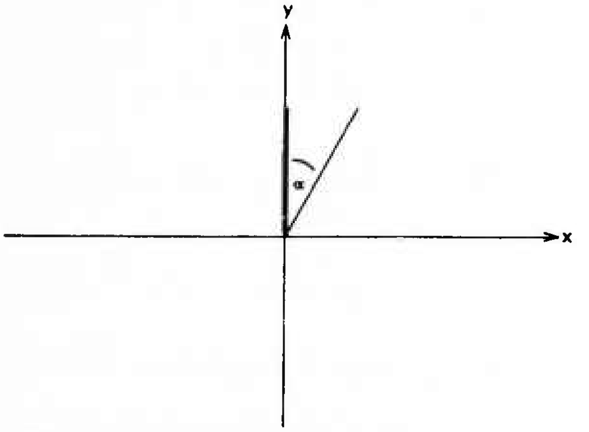


Figure 7: Simplest Kanade hyperbola: image.

The resultant constraint equation is still a hyperbola, but it is extremely simple: it is $p = -\cot\alpha(q+1/q)$, with p now a *one-to-one function* of q. As shown in Figure 8, this constraint is uniquely intercepted by the line of vertical surfaces, $q = -1/q_h$, for any value of q_h. Thus, under orthographic conditions the gradient of the generated vertical surface is uniquely defined. (Note that if the vertical line is an object edge and the horizontal line is the edge's shadow, then this gradient constrains the direction of the illuminant: see [Shafer 83b].)

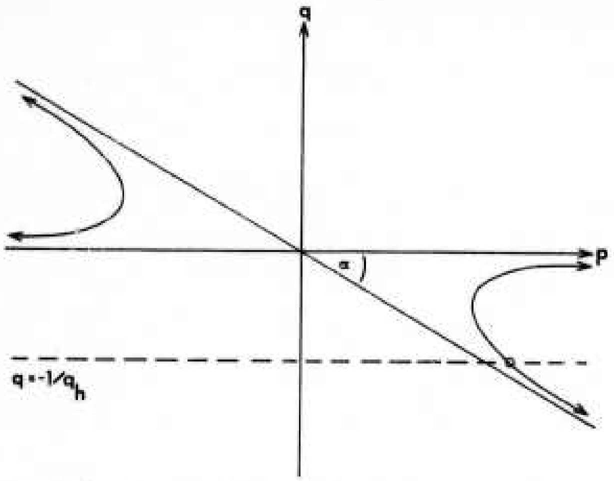


Figure 8: Simplest Kanade hyperbola: gradient space.

This special case hyperbola has several interesting properties. The first is that the minimum value of p always occurs at $q = -1$, *independent* of α. Since in orthographic photographs there is no indication of the sensor tilt, q_h, the observer is free to select a tilt at will. The choice of $q_h = -1$ guarantees that left-right slant is minimized (i.e. the surface "regresses to the frontal plane"). This value of q is equivalent to looking down at the horizontal plane at 45°; this angle is commonly used in architectural drawing [Morgan 50].

The second property is that under pure orthography all right angles with a vertical side behave identically, in one respect. Distances on the horizontal plane on which they stand can be read off *from the image*, independently of the angle α that their images form. Consider Figure 9. Let the vertex be at relative depth $z = 0$; distance increases towards the observer. Draw the retinally horizontal line $v = \cot\alpha$; the segment intercepted by the angle is of length 1. The total depth at the left intercept is $z = \cot\alpha/q_h$, since the vertical line has no p component and it increases in depth proportionally to the the sensor tilt. The total depth at the right intercept is the depth at the left *plus* the pure p component depth increase due to a movement of 1 image unit to the right. Thus, the depth at the right is $\cot\alpha/q_h - \cot\alpha(q_h+1/q_h) = -\cot\alpha q_h$, using the function relating p to q_h.

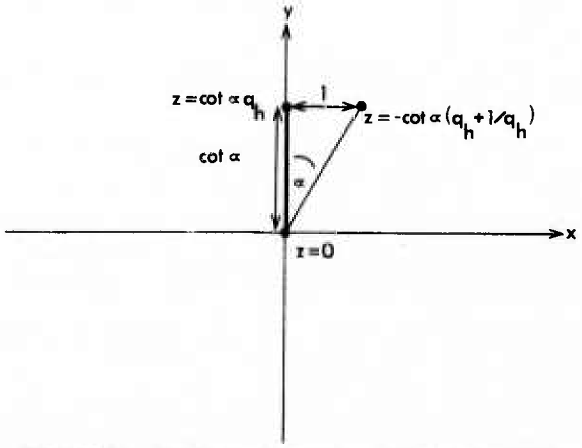


Figure 9: Right angle depths: image calculations.

This can all be summarized by stating that on any line $y = c$, the depth on the vertical leg is c/q_h and the depth on the horizontal leg is cq_h, *independent* of α. In particular, any rectangular prism of whatever size, resting on a horizontal surface with known tilt, can be easily labeled for relative depth in the image itself, starting at any vertex and propagating depth changes outwards: see Figure 10.

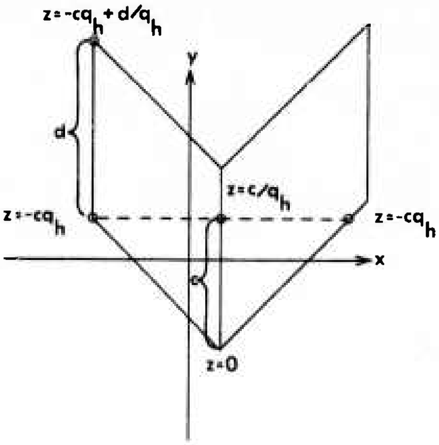


Figure 10: Right angle depths: propagation.

(An alternate derivation is shown in the side view of Figure 11. Along the plane of constant depth, at c units above the vertex, the environmental vertical has depth change c/q_h by similar triangles. Similarly, the horizontal's is cq_h. This side view also indicates the independence of depth calculations with respect to the image of the right angle; all that matters is the relative height in the image plane, and the environmental labels of vertical line or horizontal plane.)

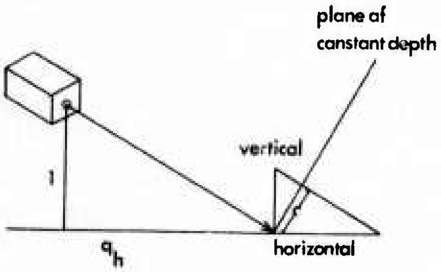


Figure 11: Right angle depths: side view.

8 Ambiguity in Labeling

In the previous section, we gave algorithms for exploiting the perpendicularity that arises between horizontal and vertical lines. We demonstrate here that that configuration's power comes not from the perpendicularity per se, nor even from the fact that the surface that is formed is vertical, but from their individual environmental labels. We show this by demonstrating that two general perpendicular lines even within a environmentally vertical plane, give rise to ambiguous surface orientations. In this discussion, we make the simplest of assumptions: orthographic imaging with known sensor parameters and no roll; basically, this is a counter-example.

Consider Figure 12; it is the image of an environmentally vertical plane in which there is embedded a right angle. Neither side of the angle is environmentally horizontal or vertical, however. The constraint in the gradient space that the image generates, from the assumption that it is environmentally right, is again the Kanade hyperbola: see Figure 13. However, because of the orientation of the image angle, this hyperbola is no longer a function. Further, some values of q have no corresponding p; certain lines of vertical surfaces would not intersect this constraint curve. This is another way of saying that some values of sensor tilt q_h are incompatible with the interpretation of the image angle as a right angle in a vertical plane. (This was not so in the case with environmentally labeled sides.) Worse, nearly every line of vertical surfaces that does intersect it intersects it twice. That is, for nearly every sensor tilt for which the image has an interpretation as a vertical plane, it has two possible gradients.

It turns out that the two interpretations are somewhat difficult to visualize. Perhaps the best way to view them is with a physical construction, rather then by studying Figure 14. The case where the line of verticals is tangent to the hyperbola probably corresponds to the configuration where both legs are at 45^o to the environmental horizontal.

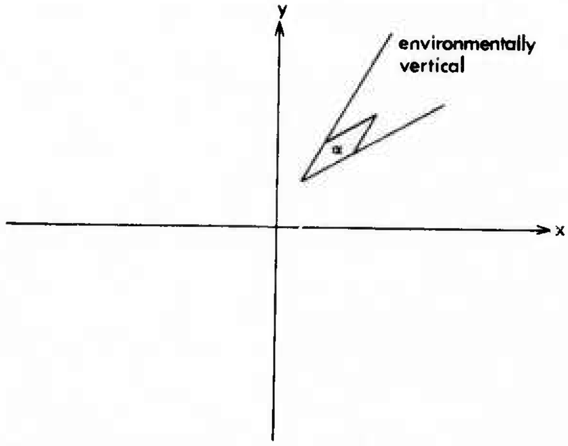


Figure 12: Ambiguous vertical planes: image.

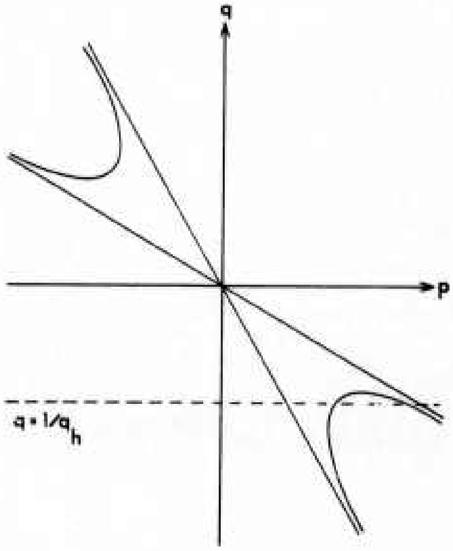


Figure 13: Ambiguous vertical planes: gradient space.

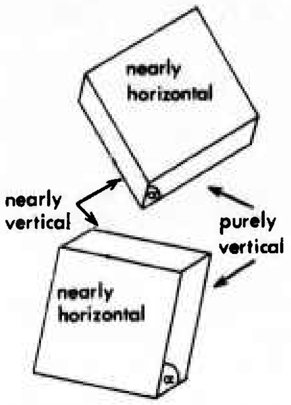


Figure 14: Ambiguous vertical planes: interpretations.

9 Discussion: Meta-heuristics

Although some of the relationships discussed in this paper have been absolute, many of them depended on heuristic assumptions. Most of the assumptions were of a similar form. The basic reasoning was as follows.

Certain preferred environmental objects create specific image configurations; for example, the images of environmentally vertical lines converge to a vanishing point. However, other environmental objects could also create the same configuration; for example, environmentally horizontal or oblique lines could also converge on the same vanishing point. The heuristics throughout assumed that the image configurations could be uniquely inverted as to cause: here, convergence implies environmentally vertical. More simply, the presence of an image feature similar to a preferred object's image features was taken as evidence for the preferred object. This "near implies preferred" meta-heuristic has proven useful in several other contexts, specifically shape from texture and skewed symmetry.

What sorts of environmental objects are preferred? One basis for preference is the simplicity with which image signatures can be inverted. For example, in the gradient space, both horizontal and vertical surfaces are easy to manipulate because their classes are small and well-defined; oblique surfaces are not. Horizontal and vertical surfaces are therefore preferred. This meta-heuristic that "preferred implies simple" has also proven useful in other contexts.

But perhaps the most evidence of the utility of the meta-heuristics is in the suggestion that foveation serves purposes other than an increase in resolution. Viewing perpendicularities off-axis under perspective leads to difficult mathematics; foveating them makes the math very simple. Thus, foveation helps to create simple signatures and helps define a preferred object. The implication for image understanding might be that all near-perpendicularities should be foveated; they might be the images of an easily determinable local vertical surfaces.

Acknowledgements

I thank Kerny Calaway for her graphic skills.

References

[Fischler 81] Fischler, M.A., and Bolles, R.C. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." *Comm. ACM 24*, 6 (June 1981), 381-395.

[Herman 82] Herman, M., Kanade, T., and Kuroe, S. Incremental Acquisition of a Three-Dimensional Scene Model from Images. Proceedings of the ARPA Image Understanding Workshop, Sept., 1982.

[Horn 77] Horn, B.K.P. "Understanding Image Intensities." *Artificial Intelligence 8*, 2 (April 1977), 201-231.

[Horn 82] Horn, B.K.P. Sequins and Quills--Representations for Surface Topography. In *Representation of Three-Dimensional Objects*, Bajcsy, R., Ed.,Springer Verlag, 1982.

[Ikeuchi 80] Ikeuchi, K. Numerical Shape from Shading and Occluding Contours in a Single View. AI Lab Memo 566, MIT Artificial Intelligence Lab., Feb., 1980.

[Kanade 79] Kanade, T. Recovery of the 3-Dimensional Shape of an Object from a Single View. Technical Reptort CMU-CS-79-153, Carnegie-Mellon University Computer Science Department, Oct., 1979.

[Kender 79] Kender, J. R. Shape from Texture: An Aggregation Transform that Maps a Class of Textures into Surface Orientation. Proceedings of the Sixth International Joint Conference on Artificial Intelligence, Aug., 1979, pp. 475-480.

[Kender 80a] Kender, J.R., and Kanade, T. Mapping Image Properties into Shape Constraints: Skewed Symmetry, Affine-Transformable Patterns, and the Shape-from-Texture Paradigm. Proceedings of the First Annual National Conference on Artificial Intelligence, American Association for Artificial Intelligence, Aug., 1980, pp. 4-6.

[Kender 80b] Kender, J.R. *Shape from Texture.* Ph.D. Thesis, Carnegie-Mellon University Computer Science Department, Nov. 1980.

[Mackworth 73] Mackworth, A. K. "Interpreting Pictures of Polyhedral Scenes." *Artificial Intelligence 4*, 2 (Summer 1973), 121-137.

[Morgan 50] Morgan, S. W. *Architectural Drawing.* McGraw-Hill, New York, 1950.

[Shafer 83a] Shafer, S.A., Kanade, T., and Kender, J.R. "Gradient Space under Orthography and Perspective." *Computer Graphics and Image Processing* (To appear 1983).

[Shafer 83b] Shafer, S.A., and Kanade, T. "Using Shadows in Finding Surface Orientations." *Computer Graphics and Image Processing* (To appear 1983).

[Woodham 78] Woodham, R.J. *Reflectance Map Techniques for Analyzing Defects in Metal Castings.* Ph.D. Thesis, MIT Artificial Intelligence Lab., June 1978. Available as AI-TR-457

Adapting Optical-Flow to Measure Object Motion in Reflectance and X-ray Image Sequences

Nancy Cornelius

Department of Electrical Engineering

and

Takeo Kanade

Department of Computer Science

Carnegie-Mellon University
Pittsburgh, Pa. 15213

The authors

Abstract

This paper adapts ~~Horn and Schunck's~~ a previous *work on optical flow [3] to the problem of determining arbitrary motions of objects from 2-dimensional image sequences. The method allows for gradual changes in the way an object appears in the image sequence, and allows for flow discontinuities at object boundaries. We find velocity fields that give estimates of the velocities of objects in the image plane. These velocities are computed from a series of images using information about the spatial and temporal brightness gradients. A constraint on the smoothness of motion within an object's boundaries is used. The method can be applied to interpretation of both reflectance and x-ray images. Results are shown for models of ellipsoids undergoing expansion, as well as for an x-ray image sequence of a beating heart.*

Introduction

Interpreting the motion of objects from a sequence of images is difficult because image changes may be due to a number of factors. First, image changes may be due to object translations or rotations, or to relative motion of one object such that it occludes another. Second, changes may occur when non-rigid objects change shape or size. Third, parts of an image need not change even though they correspond to a moving object; for example, regions of an image corresponding to flat surfaces of constant reflectance may exhibit no change if the object undergoes only translation. Fourth, changes may result from motion of the observer. Thus, effective algorithms that measure object motion from sequences of images should do two things:

- They should distinguish between image changes due to motion of objects, due to deformation of objects, and due to occlusion.

- They should determine whether regions of an image that exhibit no apparent brightness changes correspond to moving surfaces.

This paper develops methods for assigning velocities to image points by examining changes in brightness at each point in a sequence of images. While many of the techniques may be applicable to environments where the observer is moving, the emphasis will be on interpreting image sequences where the observer is stationary and only objects move. In general, we must notice that motion analysis from images cannot be solved without making assumptions about the underlying motion of objects represented in the image sequence.

Horn and Schunck [3] addressed a problem of computing optical flow from an image sequence. They define optical flow as "the distribution of apparent velocities of movement of brightness patterns" in a sequence of images. Usually optical flow refers to the flow of the imaged world across the retina as a biological observer moves continuously through the world. However, if we assume a stationary viewer and assume there are no changes in the brightness patterns as a result of the motion, then Horn and Schunck's definition of optical flow gives the velocities of objects projected onto the image plane. To say that there are no changes in the brightness patterns means that the image brightness corresponding to a single physical point on an object is the same from one frame to the next. This restriction permits only translation of objects parallel to the image plane and does not allow arbitrary rotations or perspective transformations. In order to compute optical flow, Horn and Schunck assumed that the velocities varied smoothly over the entire image. This assumption has limited utility in real images where object boundaries are usually places of discontinuous velocity for both the case of a moving object and for an observer moving with respect to a static scene.

Our approach also involves computing velocities at the points in an image, but our method differs from Horn and Schunck's in two important ways. First, the the velocity smoothness constraints are applied only within regions that are separated from the rest of the image by recognizable boundaries. Velocities are free to change abruptly across these boundaries. Second, changes in the brightness patterns are allowed so that velocities more closely represent the arbitrary motions of objects projected onto the image plane. For example, gradual shading changes that occur with rotation relative to the light source may be accommodated.

The methods developed are applied to models of ellipsoids undergoing expansion and to x-ray image sequences of a beating heart. In the latter case the pattern changes of interest are those that occur when the heart changes shape in a direction perpendicular to the image plane.

Problem Statement

If there is no a priori knowledge about the structure of objects in a scene, then measurement of velocity relies on local information about temporal and spatial gradients of image brightness. This local information provides only one constraint, the change in brightness at a given point, while the velocity of a point in an image has two components. In simple situations, where moving objects only undergo translation parallel to the image plane without changing their pattern in the image, this constraint determines the component of velocity parallel to the brightness gradient. When the brightness gradient is zero in the direction of motion (eg. flat region of an object with constant reflectance or a stripe pattern in the direction of motion), then there is no local velocity information. In all cases additional constraints must be imposed to determine the two components of velocity in the image plane as well as to determine the changes in the image pattern.

Let the image brightness projected by a point on a moving object at a time t be given by $I(x,y,t)$. At a later time $t+dt$ the same object point has moved so that its projected position in the image plane is given by $(x+dx,y+dy)$. The brightness of this point may have changed to a value $I(x+dx,y+dy,t+dt)$. Such a change occurs when lighting and shading change as an object rotates or when the object itself changes shape. The total rate of change of brightness dI/dt is given by:

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} \qquad (1)$$

where $\partial I/\partial x$ and $\partial I/\partial y$ are the x and y components of the spatial brightness gradient and $\partial I/\partial t$ is the temporal brighiness change measured at the point (x,y). The three variables that are to be determined are the x and y components of velocity, i.e. dx/dt and dy/dt, respectively, and the brightness change dI/dt. To simplify the notation, we introduce the abbreviations I_x, I_y and I_t for the partial derivatives of brightness with respect to x, y and t and the abbreviations v_x and v_y for the x and y velocity components. Equation (1) can then be rewritten in the following way:

$$\frac{dI}{dt} = I_x v_x + I_y v_y + I_t \qquad (2)$$

To solve this equation for the velocities (v_x,v_y) and the rate of brightness change (dI/dt), other constraints must be applied that restrict the allowable motions. For example, the assumption can be made that the velocity and pattern changes are constant or that they change smoothly within a region. It could also be assumed that the velocities and patterns vary in a constrained manner over time [4].

In the next section, we review Horn and Schunck's method for computing optical flow, and identify problems with it. The remaining sections propose a set of modifications and extensions to cope with those problems. First, we present a technique that permits velocity flow discontinuities at boundaries. Then we suggest a way to accommodate some of the changes in brightness patterns that occur as a result of motion. The final section presents results obtained by applying the modifications to a model of an expanding ellipsoid and an example that incorporates all of these techniques to analyze heart motion from a sequence of x-ray images.

Horn and Schunck's Method for Computing Optical Flow

Horn and Schunck [3] assumed no pattern change in the image so that the brightness change with time corresponding to a single physical point dI/dt is equal to zero, i.e.:

$$I_x v_x + I_y v_y + I_t = 0 \qquad (3)$$

This assumption severely limits the allowable motions. Rotations, translations in depth and deformations often result in changes in the image brightness pattern and violate this assumption. Horn and Schunck made the additional assumption that neighboring points have similar velocities. To implement this smoothness constraint, they constrained the local change in velocity by minimizing the square of the magnitude of the spatial gradient of the velocity components:

$$\varepsilon^2 = \left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2 \qquad (4)$$

In order to solve for the optical flow v_x and v_y, Horn and Schunck combined the two assumptions (the zero brightness change and the smoothness constraint) by minimizing the following function:

$$\int \left[\left(\frac{dI}{dt}\right)^2 + \alpha^2(\varepsilon^2)\right] dxdy \qquad (5)$$

where the integral is over the entire image and α^2 is a weighting factor that depends on the noise in the gradient measurements. The following iterative formulae provide the solution for the flow velocities that minimizes equation (5):

$$v_x^{k+1} = \bar{v}_x^{\,k} - \frac{I_x(I_x v_x^{\,k} + I_y v_y^{\,k} + I_t)}{\alpha^2 + I_x^2 + I_y^2} \qquad (6)$$

$$v_y^{k+1} = \bar{v}_y^{\,k} - \frac{I_y(I_x v_x^{\,k} + I_y v_y^{\,k} + I_t)}{\alpha^2 + I_x^2 + I_y^2}$$

where $\bar{v}_x^{\,k}$ and $\bar{v}_y^{\,k}$ denote local averages of the velocity components computed at the kth iteration. A region where there is no apparent local velocity information (eg. flat region of constant reflectance) will derive its velocity from the surrounding region, because during the iterative process, velocities will tend to propagate and fill in these regions.

There are three primary problems with this technique. The first two involve the boundaries. First, the technique does poorly when there are discontinuities in the velocity field or in the brightness gradients, because of the smoothness assumption. The discontinuities occur at object boundaries. Second, the same property that allows velocities to propagate within an object tends to extend erroneous velocities outside the area of an object. The problem is most conspicuous for the case where an object is moving against a uniform background. In this case it is not possible to distinguish the velocity of the object from the velocity assigned to the uniform background. Third, motion is constrained to be parallel to the image plane because of the assumption that an object does not change the way it appears in the image from frame to frame.

Boundary Constraints

The previous section suggests that discontinuities in velocity which occur at object boundaries must be explicitly accounted for in order to accurately determine velocities within the boundaries. We propose to allow for these discontinuities by applying the smoothness constraint separately to regions on either side of an image boundary. This can be done once the projection of the object boundaries have been located in the image. As we see next, implementation does not require that the image be segmented into regions corresponding to objects, rather only that the location of *possible* object boundaries be determined.

Image boundaries occur when one object moves in front of another: these are called occluding boundaries. Image boundaries can also occur due to the painted patterns or non-occluding edges on the object: these are non-occluding boundaries. (There are also boundaries due to object shadows, but these are not explicitly dealt with here.) In terms of motions across them, there is an important difference between occluding and non-occluding boundaries. A non-occluding boundary has consistent motion on both sides -- there is no velocity discontinuity. The regions on both sides of an occluding boundary can have different velocities. We must process velocity flow data at a boundary differently according to the type of boundary. The smoothness constraint is enforced across non-occluding boundaries, but not across occluding boundaries. This procedure permits spatial discontinuities in flow velocity to occur when one object moves in front of another.

To apply this method, we need not predetermine whether a boundary is occluding or non-occluding. First, the nearby velocities are computed based on an assumption of non-occlusion; the smoothness constraint is applied across the boundary. Next, the velocities are recalculated assuming occlusion; the smoothness constraint is not enforced across the boundary. Finally, the result that best satisfies the equation for *dI/dt* (equation (2)) and the smoothness constraint is retained. In this way, the boundary types can be locally determined without explicit segmentation of the image into object regions. This test is repeated with each iteration.

Pattern Changes

A pattern change refers to the change in image brightness of the same physical point on an object from one frame to the next. A pattern change will occur when points on the object are obscured or revealed in successive image frames. This type of change causes discontinuities in the velocity across occluding boundaries. These changes have been accommodated by the method in a previous section.

There is also another type of pattern change. For example, when an object rotates and the lighting hits the object in a different way, it results in different shading. For a Lambertian surface the shading change of a given physical point on an object is given by:

$$\frac{dI}{dt} = \frac{d}{dt}(k\ cos(\tau)) = -k\ sin(\tau)\frac{d\tau}{dt} \qquad (7)$$

where τ is the angle between the incident light and the the surface normal, and k is a constant. If the surface orientation is known, then *dI/dt* gives a measure of the change in orientation.

Here we propose to allow for such pattern changes in the image by constraining them to vary smoothly within boundaries. We can think of the pattern change (*dI/dt*) as another velocity component. While *dI/dt* is not strictly a velocity, we constrain the variations in *dI/dt* to vary smoothly within object boundaries, just as was done for the velocity components. Thus we can define a smoothness measure of change in brightness variation:

$$\varepsilon_B^2 = [\frac{\partial}{\partial x}(\frac{dI}{dt})]^2 + [\frac{\partial}{\partial y}(\frac{dI}{dt})]^2 \qquad (8)$$

New Algorithm

Now we can present our new algorithm which incorporates the considerations on boundaries and pattern changes. To summarize, this algorithm assumes: (a) the brightness changes of a single physical point can be described by the first order expansion, equation (2); (b) velocity changes in a neighborhood are similar, unless the neighborhood contains an occluding boundary; (c) the rate of pattern change (*dI/dt*) is also similar in a neighborhood. To impose these assumptions we define an error factor for each.

The first factor is the error in satisfying equation (2), i.e.:

$$\varepsilon_I^2 = (\frac{dI}{dt} - I_x v_x - I_y v_y - I_t)^2 \qquad (9)$$

Since we allow *dI/dt* to be nonzero, it is included in ε_I^2. The second error factor is a measure of the departure from a spatially smooth velocity field, ε_S^2, which is the same as equation (4). The third error factor is given by equation(8) and measures the departure from a spatially smooth pattern change.

We minimize the sum of these error factors computed over the image:

$$\text{minimize} \sum_i \sum_j \ \varepsilon_I^2(i,j) + \alpha^2 \varepsilon_S^2(i,j) + \beta^2 \varepsilon_B^2(i,j) \qquad (10)$$

An iterative form of the solution is found for the velocities at the *(k+1)* iteration in terms of the spatial and temporal brightness gradients and the neighboring velocities at the *k-th* iteration:

$$v_x^{k+1} = v_x^k - \frac{\beta^2 I_x\left[I_x v_x^k + I_y v_y^k + I_t - (dI/dt)^k\right]}{\alpha^2 + 2\alpha^4\beta^2 + \alpha^2\beta^2 I_x^2 + \alpha^2\beta^2 I_y^2} \qquad (11)$$

$$v_y^{k+1} = v_y^k - \frac{\beta^2 I_y\left[I_x v_x^k + I_y v_y^k + I_t - (dI/dt)^k\right]}{\alpha^2 + 2\alpha^4\beta^2 + \alpha^2\beta^2 I_x^2 + \alpha^2\beta^2 I_y^2}$$

$$\left(\frac{dI}{dt}\right)^{k+1} = \left(\frac{dI}{dt}\right)^k + \frac{\left[I_x v_x^k + I_y v_y^k + I_t - (dI/dt)^k\right]}{\alpha^2 + 2\alpha^4\beta^2 + \alpha^2\beta^2 I_x^2 + \alpha^2\beta^2 I_y^2}$$

where $\bar{v}_x^k$ and $\bar{v}_y^k$ denote averages of the neighboring velocities at the *k-th* iteration and $(dI/dt)^k$ denotes the average pattern change at the *k-th* iteration. This iterative procedure is applied everywhere in the image, but points in the neighborhood of a boundary are treated differently. Boundaries are located by finding zero crossings in the Laplacian of brightness [1] in each of a sequential pair of images and forming a union of such zero crossings. The size of a neighborhood is determined by the size of the region over which the smoothness constraint ε_S^2 is computed. Velocities are computed separately using points in the neighborhood on one side of the boundary and again using points in the neighborhood that span the boundary. This yields two different estimates for the velocity. The estimate that minimizes $\varepsilon_I^2 + \alpha^2\ \varepsilon_S^2 + \beta^2\ \varepsilon_B^2$ is used.

Results

Model of Expanding Ellipsoid

The algorithm described in this paper was tested with a sequence of images generated by modelling an ellipsoid that expands uniformly in all directions. The ellipsoid is assumed to have Lambertian surface properties and to be illuminated with a distant source perpendicular to the image plane. The image is resolved to 64 by 64 pixels and quantized to 256 brightness levels (see Figure 1A). The maximum velocity of any point in the image is approximately 0.5 pixels per frame. The background is uniform and therefore provides no information about motion. The actual velocity vectors for the expanding ellipsoid are shown in Figure 1B. These are the results we would like to obtain using our algorithm.

Figure 2 shows the results of applying Horn and Schunck's optical flow technique (equation 6) to the expanding ellipsoid. Here the smoothness constraint is applied across object boundaries. While the velocities are determined fairly accurately within the object, they are propagated erroneously beyond the object boundaries. The total error over the entire image is approximately 15%. When velocity discontinuities are taken into account as outlined above, a more accurate estimate of velocities is obtained as in Figure 3. We see that use of boundary information results in a clear demarcation of velocities within and without the object. The residual errors do not extend substantially beyond the boundaries of the object. The total error is 5%. However, the algorithm tends to overestimate the actual velocities in the vicinity of the boundary. Such inaccuracies are expected, because of the discontinuities in the brightness gradient that occur at the border between one object and another. One way to avoid this problem and possibly improve the flow velocity estimates throughout a region is to determine velocities at the boundaries of the region using another technique (see for example [2]). Such velocities at the boundary provides intitial conditions and remain fixed in the iterative procedure. To see this effect, the actual velocities at the boundary were supplied as initial conditions and remained fixed for the iterative procedure. The result is shown in Figure 4. The total error is less than 3%. For the case of the expanding ellipsoid, the velocities inside the boundary region were close to the correct values whether or not the initial boundary velocities were specified. However, there are probably other cases when a good initial guess of velocity at the boundary will substantially improve the velocity estimates inside the bounded region.

Application to X-ray Images

Though these techniques have been developed for objects imaged in visible light, we have begun to explore application of these techniques to x-ray images. Our goal is to use them to analyze motion of the heart from cine angiograms.

When optical flow techniques are applied to x-ray images, the results have a different meaning. At each point in an x-ray image, the brightness depends on the amount and density of the mass between the x-ray source and the film. Because brightnesses depend on object densities instead of reflectance, the velocities found by this method no longer apply to single physical points on the surfaces of objects. For simplicity, we assume that the density does not change in time and that the brightness changes therefore represent depth changes. In angiograms where radio-opaque dye is injected into the bloodstream, the primary x-ray attenuators are the dye and calcified bone. For this case, the assumption that pattern changes reflect changes in the depth of the heart is accurate since the dye filled heart is the primary source of motion.

X-ray images have two advantages over reflectance images of opaque objects. First, depth information is available. Second, objects are not totally occluded. The disadvantage is that a point in the image does not generally correspond to a single point on a single object. Thus the flow velocities take on a different meaning for x-ray images, as described above. To understand what this difference means, consider a reflectance image and an x-ray density image of the same object, an expanding sphere. (See Fig. 5.) In the reflectance image the brightness due to a single physical point on the sphere is the same in successive images, because it has the same surface orientation. Therefore, to determine velocity at a given point, we need only find a point of matching brightness that satisfies the global smoothness constraint. If we look at brightness along one dimension of the image, then the velocities are found by matching points of similar brightness in successive frames. (See Figure 5A.)

In an x-ray density image which records the z height as the brightness, such a simple matching of similar brightnesses frequently does not yield sensible velocities. In fact, there may be many points in one image for which there is no matching brightness in successive images. As shown in Figure 5B, matching points in successive frames of the x-ray image of a sphere based on similar brightness values, yields very large velocities near the densest part of the imaged sphere (i.e. center) where the actual velocities are small. A meaningful description of the motion from the density image would be obtained by taking the rate of brightness change (dI/dt) into account. This can be interpreted as a change in depth perpendicular to the image plane. (See Figure 5C.) For x-ray images of a beating heart, the brightness at a point in the image is dependent on the depth of the heart cavity perpendicular to the image plane. Thus the pattern changes will reflect the expansion or contraction movement of the heart in the direction perpendicular to the image plane.

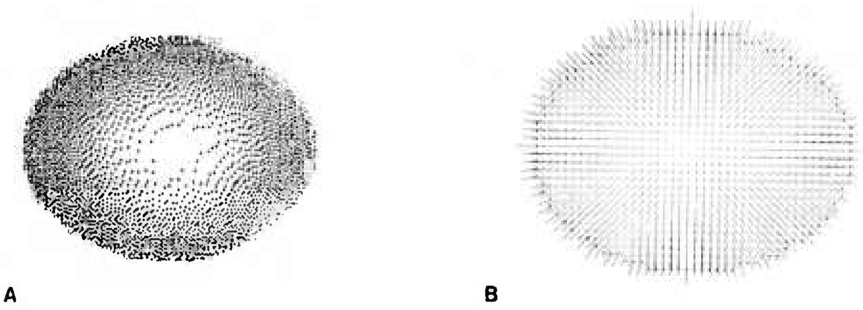


Figure 1: An image sequence was obtained by modelling an ellipsoid that expands uniformly in all directions. One frame of the sequence is shown in (A). At each point in the image we have computed the magnitude and direction of the local image velocity. The velocity vectors at each point in the image are plotted here as short line segments representing magnitude and direction. The correct velocity flow pattern determined from the model is shown in (B).

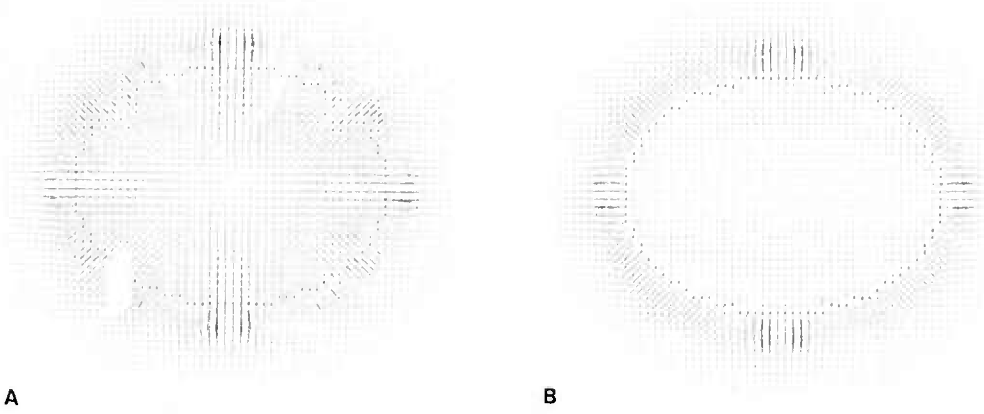


Figure 2: The velocity vectors for the expanding ellipsoid were calculated using Horn and Schunck's optical flow algorithm. The resulting flow pattern is shown in (A). No boundary constraints were imposed, so that the velocity smoothness constraint was applied across the boundaries. The result is that the velocities propagated outside the boundary. A vector plot of the velocity errors is shown in (B). Initial velocities were set to zero. The results are shown after thirty-two iterations.

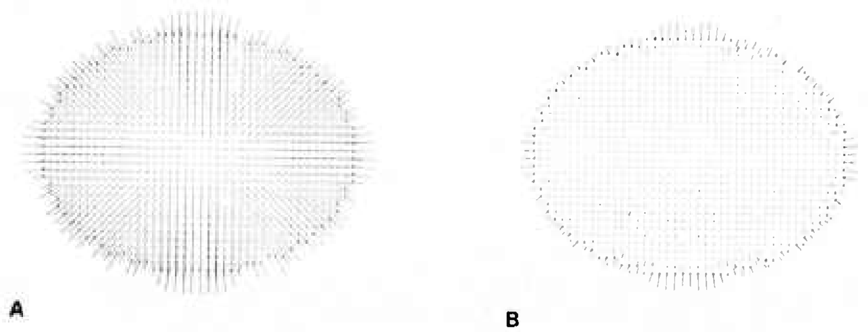


Figure 3: The velocity flow pattern in (A) was calculated for the expanding ellipsoid assuming that flow discontinuities could occur at the boundaries. The boundaries used are indicated by heavy black dots at the base of some of the velocity vectors. Velocity errors are shown in (B). The velocities computed at the boundaries are substantially greater than the actual velocities, however, the velocities inside the ellipsoid are very close to the actual velocities. Again, the initial velocities were set to zero and the results are shown after thirty-two iterations.

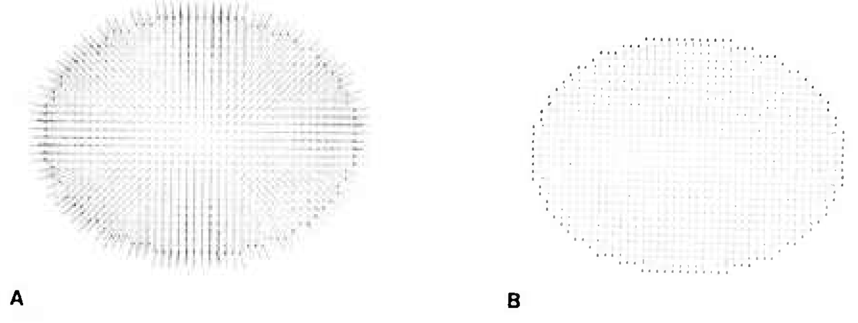


Figure 4: The velocity flow pattern in (A) was calculated for the expanding ellipsoid with the velocities at the boundaries set to the actual values. As in Figure 3, discontinuities in velocity flow were permitted at the boundary. The boundaries used are indicated by heavy black dots. The plot in (B) shows the velocity errors. The total error is less than 3% after thirty-two iterations.

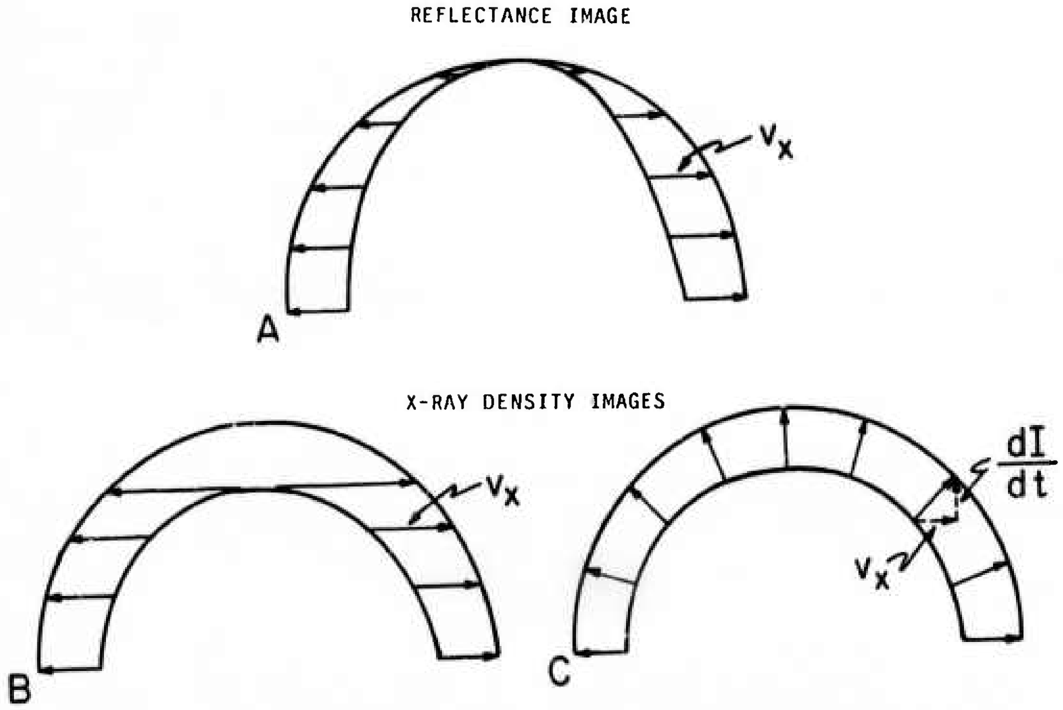


Figure 5: When a sphere expands in a reflectance image, the profile of brightnesses in a cross-section parallel to the x-axis changes as shown in (A). We assume a distant light source perpendicular to the image plane. The brightness of points on the surface do not change as the sphere expands, so surface motion can be measured by matching points of similar brightness that satisfy the smoothness constraint. When a sphere expands in an x-ray density image, the brightness of each surface point increases as shown in (B) and (C). It is no longer possible to determine velocities by matching brightnesses. We expect the velocities to be the same as for the reflectance case. However, if we simply match brightnesses as in (B),we obtain very large velocities near the center of the imaged sphere where we expect very small velocities, as well as a large flow discontinuity at the center of the sphere. If, as in (C), we allow for smoothly varying brightness changes (*dI/dt*) in addition to the motion in the plane parallel to the image, then velocities in the image plane are as expected.

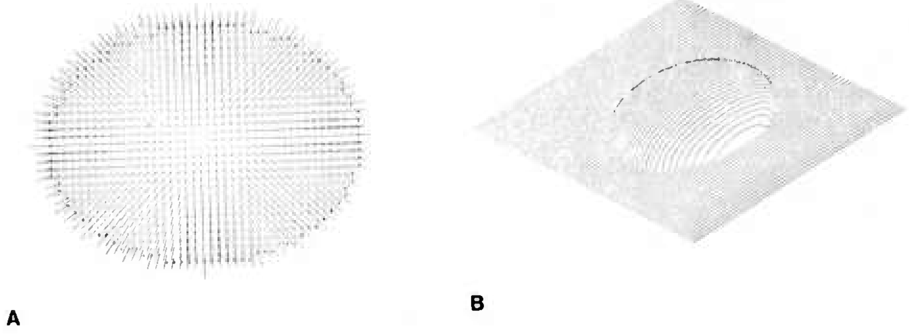


Figure 6: The expanding ellipsoid was modelled again, but the brightness at each point in the image now corresponds to the depth of the ellipsoid measured perpendicular to the image plane. The result is similar to an x-ray image of an ellipsoid. The velocities calculated from the model are shown in (A) and the rate of pattern change (*dI/dt*) is shown in (B). The pattern changes can be thought of as changes in depth.

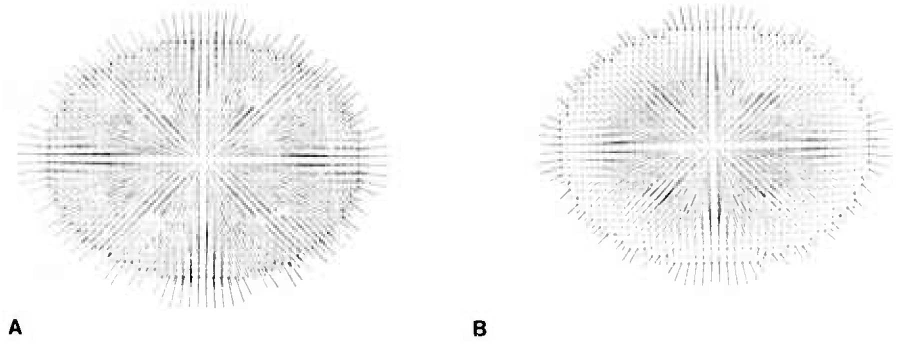


Figure 7: (A) The computed velocities for density images of the expanding ellipsoid assuming no pattern changes, i.e., $dI/dt=0$. Note the large discontinuity in velocity flow at the center of the imaged ellipsoid and the very large velocities near the center. (B) The errors in the computed velocities. Results are shown after 32 iterations.

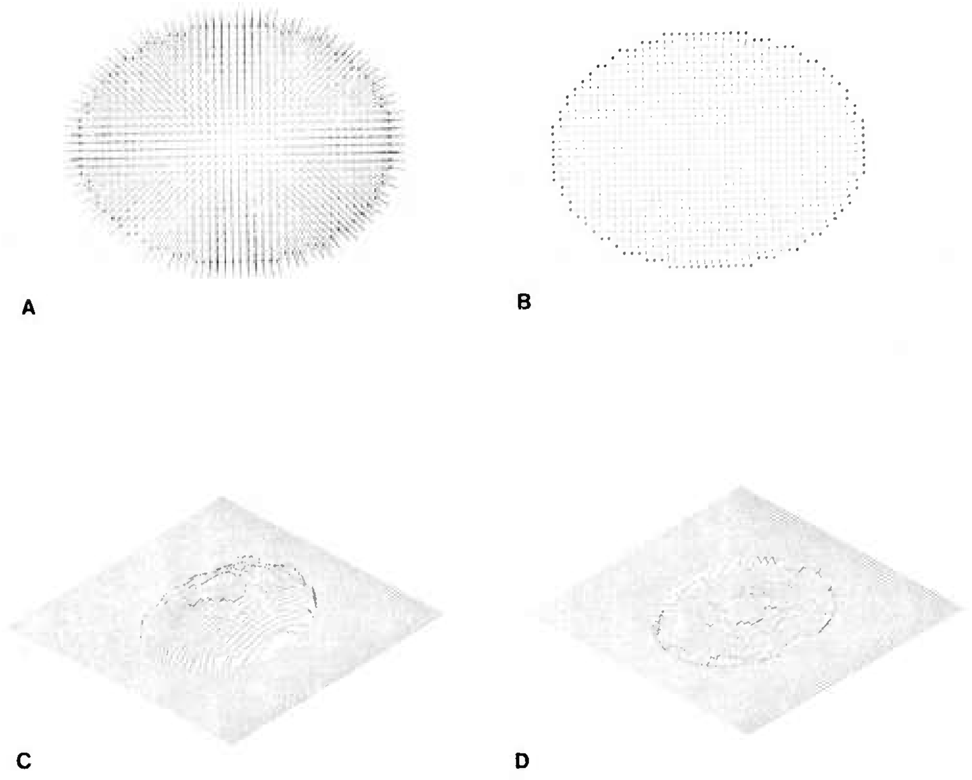


Figure 8: The velocities were computed for density images of the expanding ellipsoid allowing for velocity discontinuities at boundaries and allowing for pattern changes. The velocities were preset to the actual values at the boundaries. All other velocities were initialized to zero. The boundaries used are indicated by heavy black dots. Results are shown after 32 iterations. (A) The velocity flow pattern computed for the image plane. (B) The velocity errors in (A). (C) The computed pattern changes or depth changes. (D) The errors in (C).

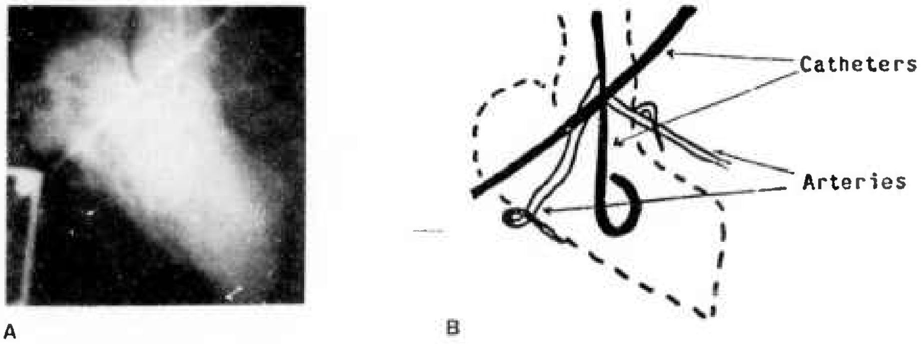


Figure 9: Experimental results were obtained for a sequence of x-ray images of a dog's heart injected with radio-opaque dye. (A) An example of a single frame in the sequence. (B) A line drawing identifying the structures.

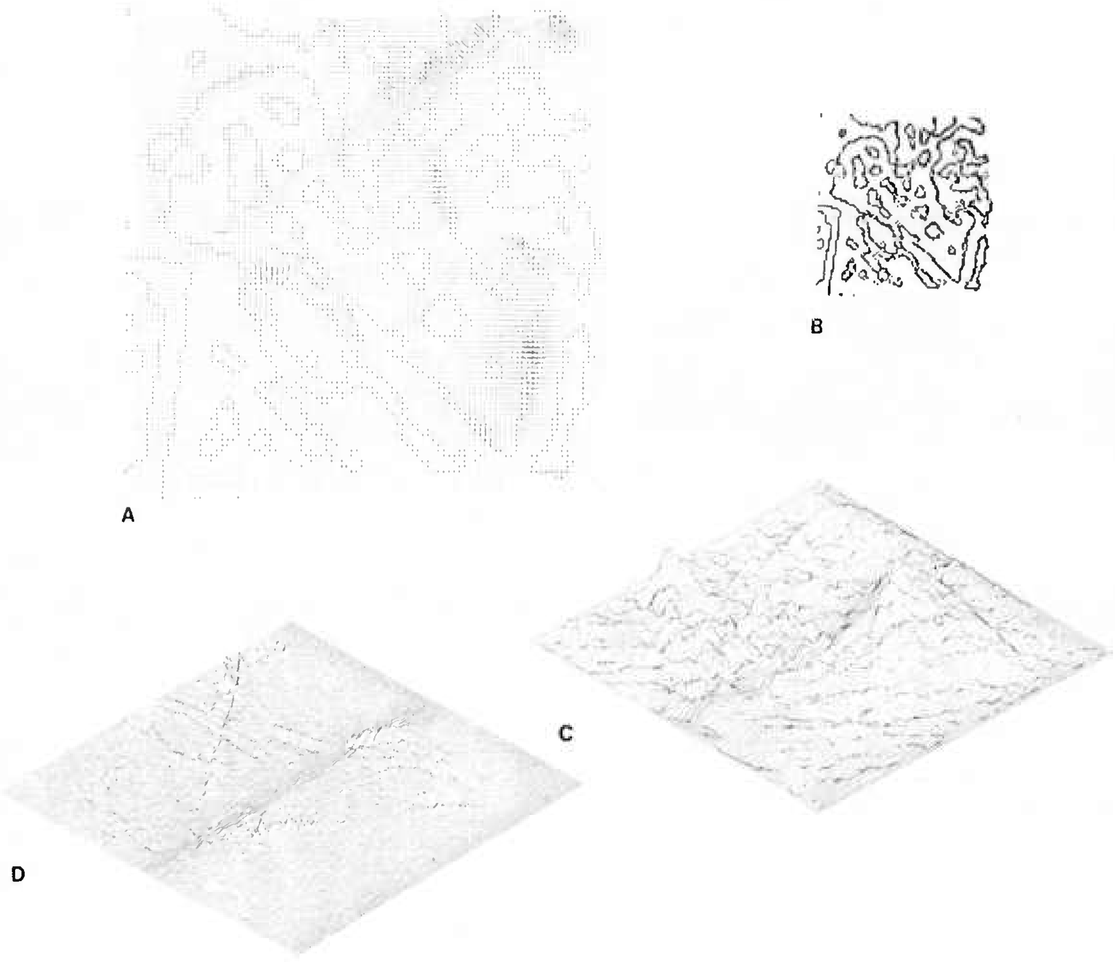


Figure 10: A sequence of x-ray images of a dog's heart were processed to obtain velocity information. Velocity discontinuities were permitted at image boundaries. (A) The velocity flow vectors. The boundaries used are indicated by heavy black dots in the figure. (B) The boundaries alone. (C) The depth changes or pattern changes. (D) The error in predicting the subsequent frame from the previous frame given the motion description in (A) and (C). The total error is less that 0.5%. Initial velocities were set to zero. The results are shown after twenty iterations.

PROCESSING RESTRICTED SENSOR MOTION

Daryl T. Lawton
Computer and Information Science Department
University of Massachusetts
Amherst, Massachusetts 01003

ABSTRACT

This paper presents

~~We~~ present a procedure for processing real world image sequences produced by relative translational motion between a sensor and environmental objects. In this procedure, the determination of the direction of sensor translation is effectively combined with the determination of the displacements of image features and environmental depth. It requires no restrictions on the direction of motion, nor the location and shape of environmental objects. It has been applied successfully to real-world image sequences from several different task domains.

The author

~~We~~ then considers several extensions and applications for such things as independently moving objects, translational blur streaks, other cases of restricted motion, computation in a hierarchical structure, and incorporation into hybrid sensor systems for autonomous navigation.

0.0 INTRODUCTION

This paper presents a procedure for processing translational motion in image sequences. The computation robustly combines the determination of the translational motion parameters, image displacements, and environmental depth. It can be used as a basic component for visually guided navigation since the other parameters of sensor motion can either be obtained using other associated devices or removed by sensor stabilization. In addition, we discuss extensions for such things as independently moving objects, translational blur streaks, and computation in hierarchical structures.

The basic procedure consists of two steps: Feature Extraction and Search. The feature extraction process finds small image areas which may correspond to distinguishing parts of environmental objects. The direction of translational motion is then found by a search which determines the image displacement paths along which a measure of feature mismatch is minimized for a set of features. The correct direction of translation will minimize this error measure and also determine the corresponding image displacements for the extracted features.

The feature extraction process finds distinctive points which are positioned at points of high curvature along contours determined by simple processes such as thresholding, zero-crossing extraction and simple local contrast measurements. Particular forms of the feature extraction process can lead to effective and very rapid implementation on proposed image processing architectures.

The search process minimizes an error measure defined with respect to a unit sphere with each point on the sphere corresponding to a different direction of sensor translation. A given direction of translation constrains the motion of extracted image features to straight lines which radiate from or converge onto a single point in the image plane. The error measure thus associates with a point on the unit sphere, corresponding to a particular translational axis, a number describing the total extent of feature mismatch along the displacement paths determined by the translational axis. Experiments have shown this error measure to be smooth and with a distinct minimum in a large neighboorhood about the correct translational axis. Because of this, the search process can be quite simple.

Experiments with several different image sequences indicate that the procedure is very robust and applicable to a wide range of real world situations. We also review particular extensions for implementing the procedure in a hierarchical computational framework, dealing with independently translating objects, translational blur-streaks, and implications for autonomous navigation.

This work was supported by grant number N00014-82-K-0464 from DARPA.

Model of X-ray Image of Expanding Ellipsoid

We show an example for a sequence of images generated by modelling an ellipsoid that expands with time. The brightness is proportional to the depth of the ellipsoid perpendicular to the image plane. The ellipsoid is expanding in all directions so that the size and brightness change as a function of time. The velocities projected on the image plane are the same as for the case of the reflectance image. We expect the brightness changes to be proportional to the actual brightness or depth. Figure 6 shows these anticipated velocities and brightness changes. Figure 7 shows the velocity field which is computed by the Horn and Schunck method (i.e., with the assumption that there are no pattern changes. $dI/dt=0$). As expected, we obtain a large flow discontinuity at the center of the image of the ellipsoid. Figure 8 shows the result of our method in which pattern changes are allowed. The velocities and pattern changes are very close to the expected results

Experimental Results for Heart Images

We have applied the methods described in this paper to x-ray images of a dog's heart taken on film at 60 frames a second. Figure 9 shows an example of a single frame of the cine angiogram. A radio-opaque dye was injected into the pulmonary artery just before the image sequence was taken. The dye can be seen filling the left ventricle, the aorta and some of the coronary arteries. The other obvious structures in the images are a couple of catheters left over from some previous injections. The film was digitized with 8 bits per pixel and resolved to 100 x 100 pixels.

The velocities were computed using equations (11). Pattern changes caused by the expansion and contraction of the heart perpendicular to the image plane were permitted and discontinuities in the velocity flow were accommodated at image boundaries as described above. The image boundaries were located at the zero crossings of the Laplacian of a smoothed version of a pair of sequential images. The computed velocities are shown in Figure 10. To verify these results, the computed motion description is used to predict the brightness in a subsequent image from the brightness in the previous image. A comparison of the predicted and actual images shows an error of less than 0.5%. While this does not show that the motion description is actually a good one, it does show that the algorithm is working as expected. In order to get a subjective opinion of the validity of the motion description, we have generated a movie of the velocity vectors for an entire heart cycle and shown that it coincides well with the apparent motion seen in the actual cine angiogram. While the motion description obtained from the analysis of x-ray images may be useful, it does not provide explicit information about motion of object surfaces. This sort of information might be obtained by using additional views of the object from different angles, or by considering a priori information about the object's shape or symmetry.

Summary

This paper extends the work of Horn and Schunck on optical flow. Their velocity smoothness constraint is relaxed at boundaries to permit discontinuities in estimated velocity where there are occluding boundaries. It is not necessary to segment the image into objects in order to use boundary information. Rather it is only necessary to locate possible boundaries which can be done by locating zero crossings in the Laplacian of the smoothed image brightness. Images of an expanding ellipsoid were used to test the resulting iterative algorithm. The results showed that discontinuities in the velocity flow could be accommodated, but that the velocity at the actual boundary may be inaccurate. It is possible to estimate the velocities at the boundary using another technique and then to use these estimates as input to the iterative algorithm.

Though the techniques were originally developed for reflectance images, we have begun to apply them to x-ray density images. To do this it has been necessary to relax Horn and Schunck's restriction that the appearance of an object not change from one image to the next. This was done by assuming that the pattern changes in the image vary gradually within object boundaries. Velocity flow patterns for x-ray images of the expanding ellipsoid were obtained in this way. The results showed that the computed velocities parallel to the image plane were very close to those obtained for reflectance images and the pattern changes were very nearly proportional to the velocities perpendicular to the image plane. The technique was also used to analyze x-ray cine angiograms of a beating heart and produced a subjectively good description of the motion.

Acknowledgements

We would like to thank Bob Selzer and the Biomedical Imaging Lab at the Jet Propulsion Labs in Pasadena, California for providing us with the digitized cine angiogram data.

References

[1] Hildreth, Ellen C.
Implementation of a Theory of Edge Detection.
Technical Report AI-TR-579, Artificial Intelligence Laboratory, MIT, April, 1980.

[2] Hildreth, Ellen C.
The Integration of Motion Information Along Contours.
In *Proceedings of the Workshop on Computer Vision Representation and Control*, pages 83-91. IEEE, August, 1982.

[3] Horn, Berthold B. K. and Schunck, Brian G.
Determining Optical Flow.
Artificial Intelligence 17:185-203, 1981.

[4] Yachida, Nasahiko.
Determining Velocity Map by 3-D Iterative Estimation.
In *Proceedings of Seventh IJCAI*, pages 716-718. Vancouver, B.C., Canada, August, 1981.

0.1 Coordinate System

The camera model referred to through out this paper consists of a planar retina embedded in a three-dimensional Cartesian coordinate system (X,Y,Z), with the origin at the focal point and the optical axis aligned with the positive Z-axis (figure 1). The X and Y axes correspond to the gravitationally intuitive horizontal and vertical directions respectively. The image plane is parallel to the XY plane and at some distance along the Z axis. Positions in the image plane are described using a 2-d coordinate system aligned with the X and Y axes of the camera coordinate system and with the origin determined by the intersection of the image plane and the Z-axis.

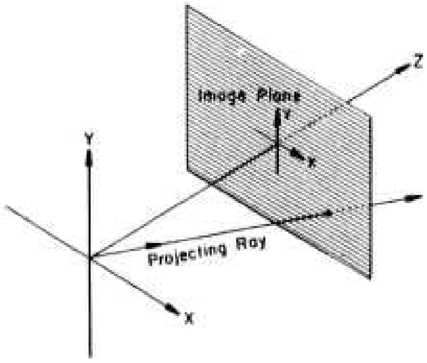


Figure 1

0.2. Translational Motion Properties

It is useful to have a set of terms for describing the motion of features in an image sequence and the corresponding motion of environmental points. We define an Image Displacement Vector to be a two-dimensional vector describing the displacement of an image feature from one image to the next. An Image Displacement Field is the set of image displacement vectors for successive images. An Image Displacement Sequence indicates the positions of an image feature over several successive images. Though we are dealing with discrete image sequences, it is often possible to descibe the continuous curve along which an image feature point is moving. This curve is called the Image Displacement Path.

Corresponding to image motions we use a set of terms for describing environmental motions. An Environmental Displacement Field is the set of three-dimensional vectors indicating the positions of environmental points at successive instants. An Environmental Displacement Sequence indicates the position of an environmental point over several instants. An Environmental Displacement Path describes the three-dimensional curve that environmental points are moving along for particular motions.

For general camera motion, there are 5 parameters [PRA81] that can be recovered from processing image motion without knowing absolute camera displacement or velocity (since absolute depth is lost): two parameters for the unit vector (T1(t), T2(t)) which describes the axis of translational motion at time t; two parameters for the unit vector (R1(t), R2(t)) describing the axis of rotation at time t; and one parameter R3(t) which describes the extent of rotation about this axis at time t. Both of these axes are positioned at the origin of the camera coordinate system. The problem of processing image motion resulting from rigid body camera motion can be organized into subcases of increasing complexity, corresponding to the number of camera motion parameters that are unconstrained.

For purely translational motion, the image displacement paths are determined by the intersection of the translational axis with the image plane. If the translational axis intersects the image plane on the positive half of the axis, the point of intersection is called a Focus of Expansion (FOE) and the image motion is along straight lines radiating from it. This corresponds to camera motion towards environmental points. If the translational axis intersects the image plane on the negative half of the axis, the point is called a Focus of Contraction (FOC) and the image displacement paths are along straight lines converging towards it. This corresponds to camera motion away from environmental points. The intersections of axes parallel to the image plane are points at infinity and are treated as FOEs.

Given the direction of translation and image displacements, the relative depths of points can be computed by solving the inverse perspective transform. Relative depth can also be inferred from the position of a feature and the extent of its displacement relative to an FOE or an FOC. This relation is expressed as

$$1) \qquad \frac{D}{\Delta D} = \frac{Z}{\Delta Z}$$

where Z is the value of the Z component of an environmental point at time t+1, del Z is the extent of environmental displacement along the Z axis from time t to time t+1, D is the distance of the corresponding image point from the FOE or FOC at time t and del D is the image point's displacement from time t to time t+1. Thus, the Z value of an environmental point can be recovered from image measurements in units of del Z, or what has been termed Time-Until-Contact by Lee [LEE80].

The set of all possible translational axes describes a unit sphere called the translational direction sphere. The procedures below are defined with respect to this sphere, rather than the image plane itself, for reasons described below.

1.0 EXTRACTION OF INTERESTING POINTS

The feature extraction process is used to determine small areas (sometimes called image points or features) in an image that are distinct from neighboring areas. This distinctiveness limits the potential matches of these image areas, and possibly reflects a correspondence to actual and significant points in the environment, such as points of high curvature on object boundaries, texture elements, surface markings, etc. (However some features, termed false features, will result from noise, occlusion, and light source effects and have behavior which is currently difficult to interpret). Features can be represented either as arrays of numbers extracted directly from an image, or as parameterized tokens describing local image properties. In this paper, we refer to features exclusively as small arrays of data values centered at some point in an image at some time t.

Following Moravec [MOR77,MOR80], the method of feature extraction used here is based upon finding image areas which are significantly different than their neighboring areas. Using a correlation measure bounded between 1 (for perfect correlation) and 0, the distinctiveness of a feature is 1 minus the best correlation value obtained when the feature is correlated with its immediately neighboring areas. Good features can then be selected by finding the local maxima in the values of the distinctiveness measure over an image.

We have extended this approach somewhat by constraining the neighborhoods over which the features are selected to contours determined by other global processes, such as zero-crossing extraction and thresholding, which are sensitive to edges.

1.1 Feature Extraction Using Zero-Crossings

The use of zero-crossings to determine significant image contours at different levels of resolution has been proposed and extensively studied by Marr et. al. [HIL80,MAR80]. In this processing an image is convolved with Gaussian-Laplacian masks ((del)**2g) of different positive widths and thresholded at zero to determine zero-crossing contours. These contours are significant since they correspond to the points of greatest change in the convolved image. The distinctiveness measure can be applied to points along these contours in the convolved image, with the local maxima determining the position of potential features. This generally has the effect of finding points of high curvature along the zero-crossing contour, although points apparently corresponding to local occlusion vertices and weak maxima will also be extracted.

Many of the weak features which are local maxima of distinctiveness can be removed by suppressing those which are at points of low curvature along the zero-crossing contours. For features which are local distinctiveness maxima, we approximate the curvature along the contour by the inner product of the normalized vectors describing the relative positions of adjacent local maxima along the contour (figure 2). These values are then thresholded between 1.0 (corresponding to high curvature) and -1.0 (corresponding to low curvature).

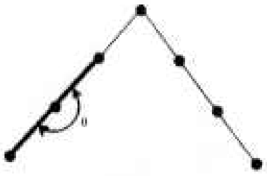


Figure 2

Use of zero-crossing based features requires specification of the sizes of the convolution masks that are employed, and deciding whether to position extracted feature points with respect to the unprocessed image or the convolved images. It is usually beneficial to use masks of various widths for sensitivity to features at different levels of resolution. The processing described below can be applied independently to the pairs of successive images formed by convolving the successive images with two such masks. Alternatively, features can be extracted from the original, unfiltered image at the positions where features were determined in the convolved images, though experience with large masks has shown that this approach can position features significant distances from their apparent position in the original image.

The images in figure 3a and figure 3b were taken from a gyroscopically stabilized movie camera held by a passenger in a car travelling down a country road in Massachusetts. They are 128x128 pixel images with 6 bits of resolution in intensity and will be referred to as the roadsign images. Figure 3c shows the zero-crossings extracted from the initial roadsign image using a (del)**2g mask with a width of 5 pixels. The distinctiveness values were computed using features which were 5x5 pixel arrays extracted from the convolved image and centered on pixels which were adjacent to the zero-crossing contour and of positive value. These features were correlated, using Moravec's norm (see below), with their 8 immediately neighboring features. The distinctiveness measure for a feature was set to 1 minus the best correlation obtained in its neighborhood, excluding itself. Figure 3d shows the local maxima in the distinctiveness measure positioned with respect to the zero-crossing contour. Figure 3e shows the results of suppressing low-curvature points using a threshold set to -0.8 (corresponding to an angle of 143.13 degrees).

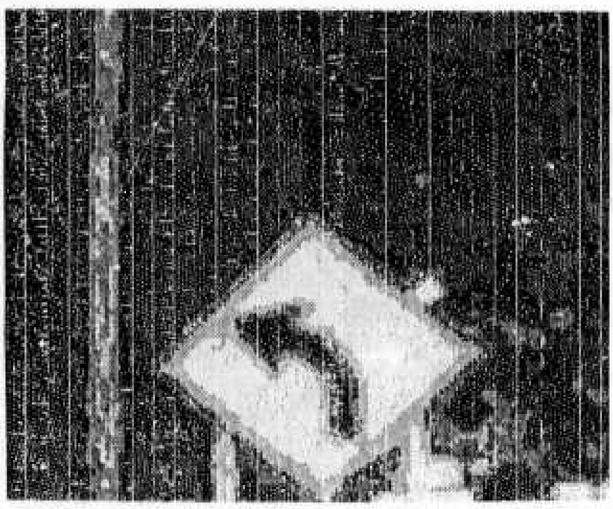


Figure 3a

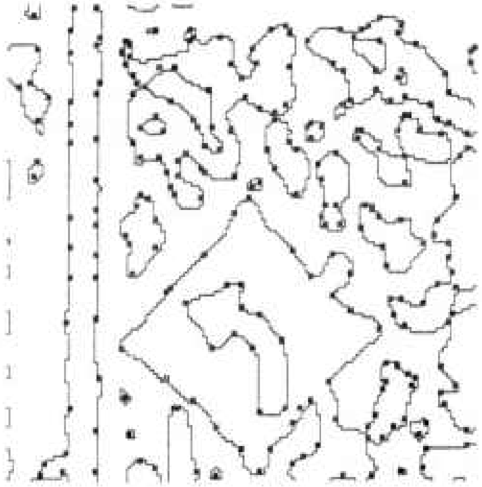


Figure 3d

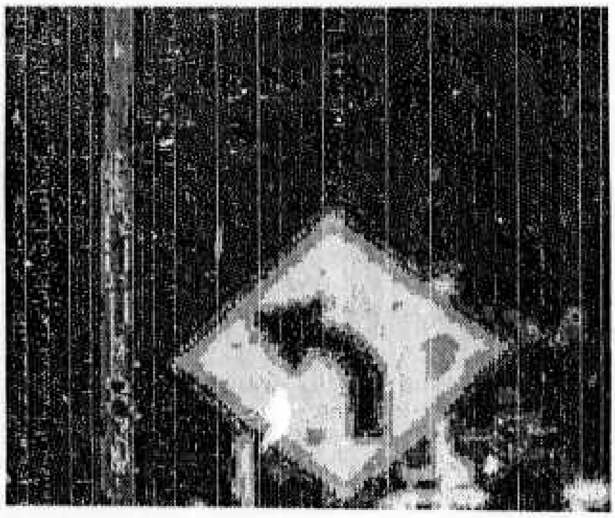


Figure 3b

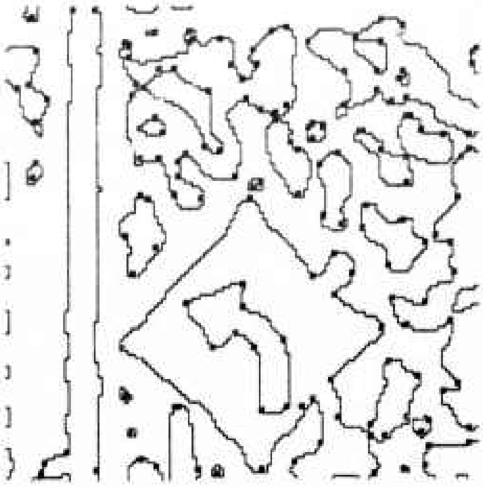


Figure 3e

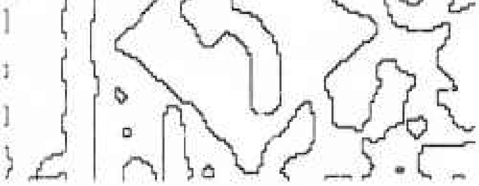


Figure 3c

Other types of contour extraction can be used besides zero-crossings and total image thresholds. A simple one is to constrain the extraction of interesting points to positions where image contrast exceeds some minimal value. Another is to use contours determined by local application of histogram guided thresholding and segmentation. This resolves many of the problems associated with using a single threshold determined for image subparts with significantly different brightnesses.

1.2 Simplifications and Speed-Ups

In image-processing architectures We have investigated, locality of processing leads to the most efficient processing. In the procedure here, the contour walking is significantly non-local. Additionally, due to the robustness of the translational processing with respect to weak features or a small number of features (see below), such optimal interesting points are not necessary.

A simple alternative to the contour walking is to use a threshold on the distinctiveness measures, with or without the determination of local maximas in distinctiveness. Examinatior of the local maximas along the telephone pole in figure 3d, reveals that these are local maximas with very small distinctiveness measures. This has been observed in general.

An additional speed-up can be obtained when features are selected from contours determined by segmentation procedures (such as thresholding or zero-crossing extraction) which produce binary images where pixel values may be represented by 1 or -1. In this case there is no need to normalize the correlation measure used to determine distinctiveness (due to constant image energy [DUD73] when each image area is stored as a -1 or 1). When such distinctiveness measures are used with a threshold on distinctiveness and local maximal extraction, very rapid rates of feature extraction can be achieved in particular architectures, on the order of a fraction of a millisecond [LAW83]. Figure 4b shows the interesting points extracted from the binary image in 4a using a threshold on distinctiveness set to 0.1 followed by local maxima extraction. The results are quite reasonable although adjacent regions can influence the extraction of features on distinct contours and points of limited curvature can be extracted. This could be remedied by restricting the calculation of distinctiveness for points only along contours of the same region (which would then require the initial determination of region labels).

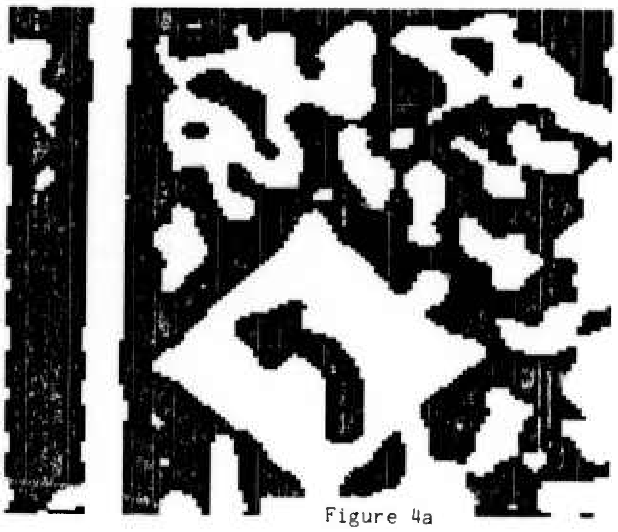


Figure 4a

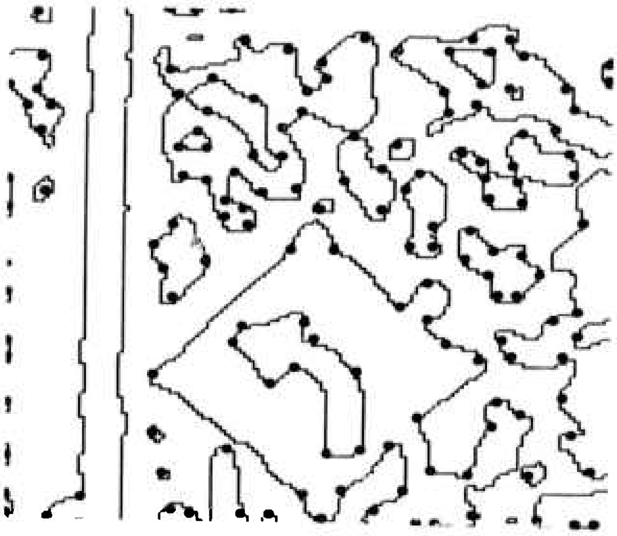


Figure 4b

2.0 DETERMINING THE AXIS OF TRANSLATION

Using the extracted features, the search process minimizes an error measure which reflects their extent of mismatch along the image displacement paths determined by a hypothesized translational axis. For example, figure 5 shows an FOE determined by a potential translational axis and the corresponding image displacement paths for some extracted features. Also shown is the match profile for a particular feature along a segment of its displacement path with respect to features positioned in the succeeding image. The adequacy of a translational axis is measured by finding the best match for each feature along the image displacement path determined by the translational axis and then summing the extent of error in these best matches.

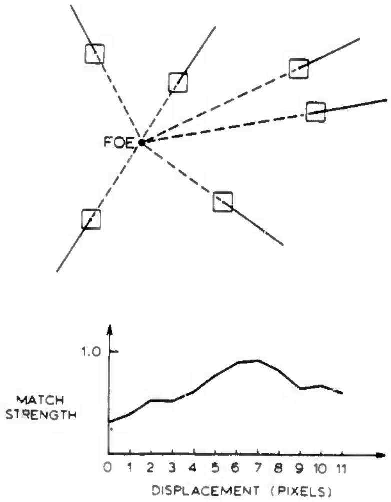


Figure 5

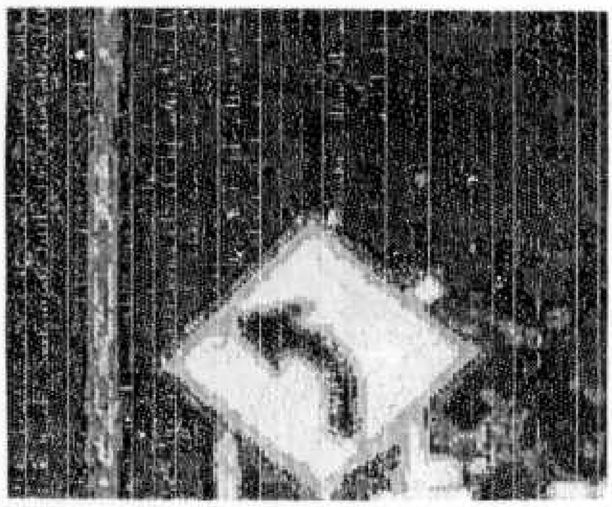


Figure 3a

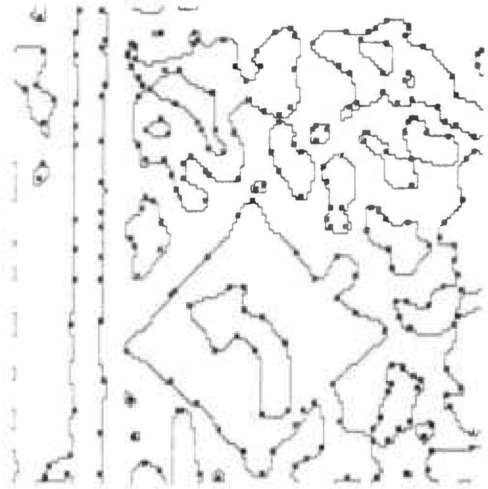


Figure 3d

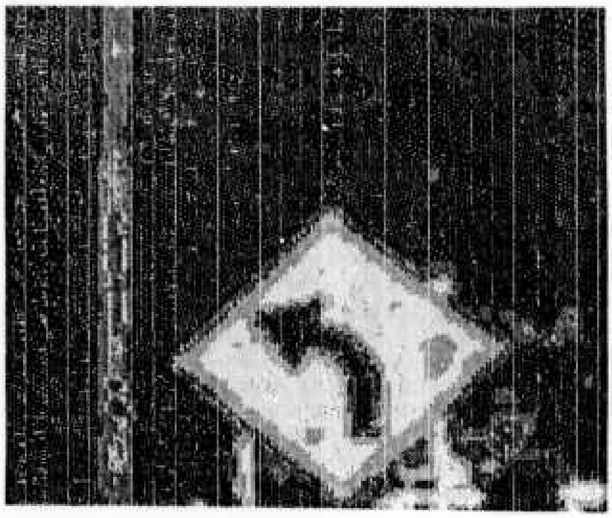


Figure 3b

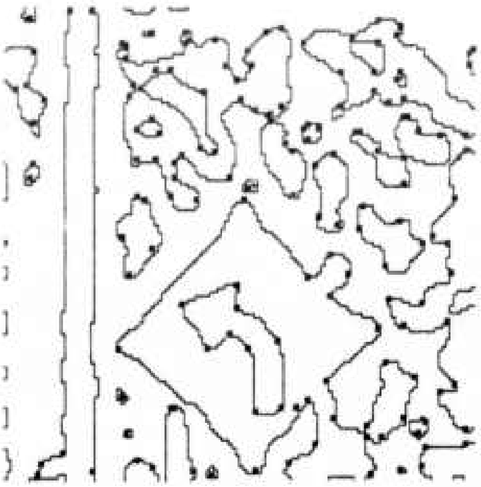


Figure 3e

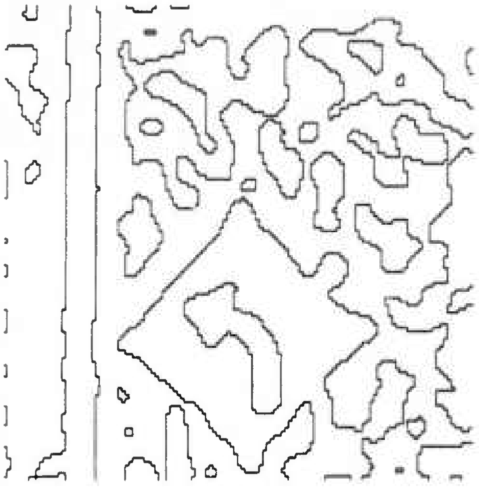


Figure 3c

Other types of contour extraction can be used besides zero-crossings and total image thresholds. A simple one is to constrain the extraction of interesting points to positions where image contrast exceeds some minimal value. Another is to use contours determined by local application of histogram guided thresholding and segmentation. This resolves many of the problems associated with using a single threshold determined for image subparts with significantly different brightnesses.

Developing the error measure requires a measure for the degree of match between features and an interpolation process for determining positions along an image displacement path. Each of these can be implemented in various ways with the choices generally involving a trade-off between the speed of evaluating the error measure and the precision with which the translational axis can be determined.

2.1 Match Metric

There are several metrics for similarity of nxn pixel features of the form A(i,j) and B(i,j), where i ranges from 1 to n and j ranges from 1 to n. We have utilized:

Normalized Correlation

$$2) \quad \frac{\sum_i \sum_j A(i,j)\times B(i,j)}{\sqrt{\sum_i \sum_j A(i,j)\times A(i,j)} \times \sqrt{\sum_i \sum_j B(i,j)\times B(i,j)}}$$

Moravec Correlation [MOR77]

$$3) \quad \frac{\sum_i \sum_j A(i,j)\times B(i,j)}{(((\sum_i \sum_j A(i,j)\times A(i,j)) + (\sum_i \sum_j B(i,j)\times B(i,j)))/2)}$$

Normalized Absolute Value Difference

$$4) \quad 1.0 - \left(\frac{\sum_i \sum_j \mathrm{abs}(A(i,j)-B(i,j))}{\sum_i \sum_j A(i,j) + \sum_i \sum_j B(i,j)} \right)$$

All of these measures have a value of 1 for a perfect match. Of these, the first choice is the most conventional, the second a good approximation to the first, and the third is the quickest to evaluate.

2.2 Interpolation Process

The interpolation process approximates the potential displacements of a feature from an initial image into a succeeding image. Depending on the accuracy required, positions along the image displacement path can be approximated roughly by setting the coordinates of the feature's position to the nearest integer value, or more accurately by performing a subpixel interpolation of the feature at each of a set of selected positions along the image displacement path. The basic trade-off is between speed and accuracy, with subpixel interpolation being a more expensive computation.

2.3 Error Measure

The error measure associates with a point on the direction of translation sphere a number describing the quality of feature matches along the image displacement paths determined by the corresponding translational axis. This error value is computed by first finding the best match for each feature along a segment of the image displacement path determined by the translational axis using one of the normalized match metrics above. Each of these values is then subtracted from one, and all the resulting values are added together to form an error measure. Thus, for a set of N features in an initial image, a hypothesized translational axis, and use of one of the match metrics above, the error measure is

$$5) \quad \sum_{i=1}^{n} (1.0 - \mathrm{bestmatch}(i))$$

where *bestmatch(i)* is the best match value associated with feature i along the image displacement path determined for it by the hypothesized translational axis.

The error measure was computed in two forms in the experiments below: a *fast evaluation* form and a *precise evaluation* form. The fast form uses the absolute value norm and the nearest integer approximation to determine feature position along the image displacement paths. The fast form is useful for evaluating image sequences with several extracted features to determine the rough position of the global minimum. However, the fast form is not adequate for fine determination of the translational axis because it does not vary smoothly with respect to small changes in the position of a translational axis, due to the nearest integer approximation for feature position.

The precise form of evaluation uses the Moravec norm and bi-linear interpolation. It has been found to vary smoothly with respect to small changes in the position of a translational axis.

2.4 Search Organization

The search process used here consists of two phases. A global sampling of the error measure determines the rough shape of the error surface, then a local search determines the minimum. The local search begins at the position where the minimum value was determined by the global sampling. The procedure used for the local search is steepest descent with a diminishing step-size. That is, the steepest descent procedure begins with a initial fixed step size and determines a local minimum using it. The step-size is then reduced and the procedure repeated until the step-size is at the desired resolution for the determination of the translational axis. In the experiments below the initial step-size was set to 0.1 and then reduced successively to 0.025 and 0.005 radians.

In general, the error measure has been found to be smooth, with a single minimum in a large neighborhood around the correct translational axis. Thus, the global sampling can be quite sparse or the initial step size of the local search quite large.

3.0 EXPERIMENTS

This procedure has been applied to several different image sequences under various conditions. These have included adding spurious and weak features; whether the features were sparse and scattered across the initial image or whether the features were in a limited image area; and whether the translational axis intersected the image plane in a visible portion of the image or whether it didn't. These experiments have shown that the procedure is robust in several important ways. It is resilient with respect to weak and false features. It can use a small number of features positioned across an image surface or a small number of features from a limited area of the image. And it is not affected by the orientation of the translational axis.

Results of processing the roadsign images are shown in figures 6, 7, and 8. The global search was used with the absolute value norm and nearest integer interpolation. The sampling increment corresponded to the vectors on the direction of motion sphere being separated by .314157 radian increments. The maximal image displacement was set to 10 pixels. Using features centered at the positions shown in figure 3e, the global sampling determined a minimum in the error function at the unit vector (-.80902, -.47554, .34548) on the direction of translation sphere.

The local search then used the Moravec norm and bi-linear interpolation. The determined translational axis was (-.83738, -.42043, .34933). The displacements of the feature points from figure 3d for this translational axis are shown in figure 6.

The procedure was repeated, but using features at the positions from figure 3d (those prior to low curvature suppression). This has the effect of introducing weak and false features into the computation. The translational axis extracted was (-.82909, -.42281, .36585) This is a difference of 0.01863 radians or 1.06765 degrees from that determined using the features indicated in figure 3e. Since the camera focal length was longer than 1, the angle between the determined translational axes is actually considerably less than this.

The procedure was also applied using the features from the restricted subarea shown in figure 7, corresponding to some faint tree texture. Using these features, the translational axis extracted was (-.84281, -.42928, .32465). This is a difference of 0.02677 radians or 1.53418 degrees with the translational axis determined using the feature centered at the positions indicated in figure 3e.

Given the direction of translation and image displacements, the relative environmental depths of image points can be recovered by the simple relation in equation 1. When image displacements are small, the inferred depth values can be quite erratic due to sensitivity to small numbers in the denominator in the left hand side of equation 1. For this reason, it is necessary to keep track of the image displacements over several successive images with concurrent updating of the inferred depth values. This was done using a sequence of four successive images from the roadsign sequence beginning with roadsign images 1 and 2 and using the features from image 1 at the positions in figure 3e. The position of the translational axis determined from images I(t) and I(t+1) was used as the initial value in the local search for determining the translational axis for images I(t+1) and I(t+2).

The displacements of all features along the contour in figure 3e were evaluated along the image displacement paths determined by the translational axis found for images I(1) and I(4). From these displacements the depth values for image points along the contour were computed using equation 1.

The roadsign sequence is particularly nice for presenting depth processing results because the three environmental objects in the images are at three distinct depths. This is shown in figurc 8a by the three distinct clusters in the histogram of the depth values calculated for the points along the contour. The units in the histogram are cummulative time-until-contact values. That is, the depth is given in units of the displacement of the camera from I(1) to I(4) along the Z-axis. From left to right, the first peak corresponds to the sign, the second to the pole, and the third to the trees. As can be seen, there is a wide range of depths associated with the trees. Mapping these clusters back onto contour points from figure 3c yields: the boundary shown in figure 8b (the sign), the boundary shown in figure 8c (the pole), the boundary segment shown in figure 8d (the trees). Points near the image boundary of I(1) were ignored because the processing did not take into account occlusion effects along the image boundaries.

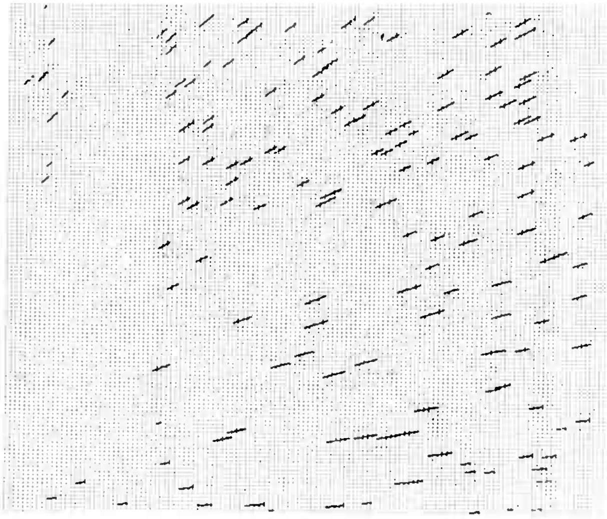


Figure 6

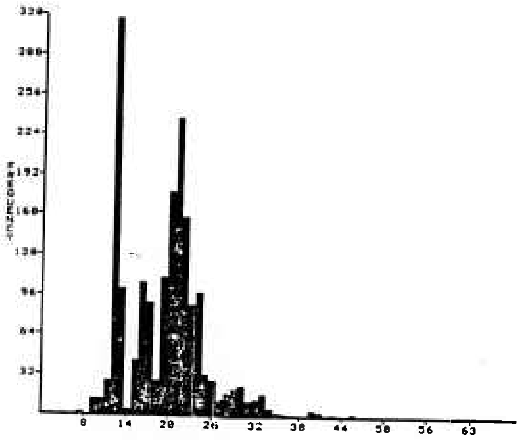


Figure 8a

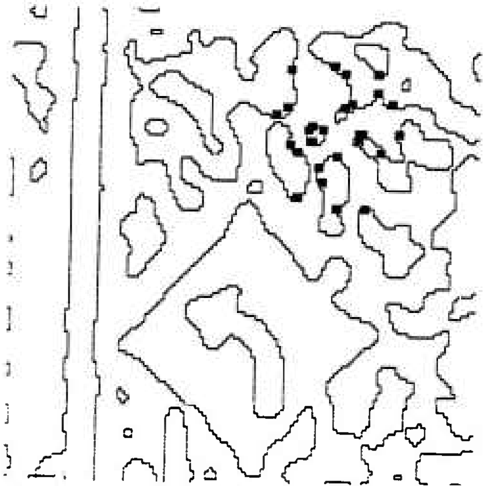


Figure 7

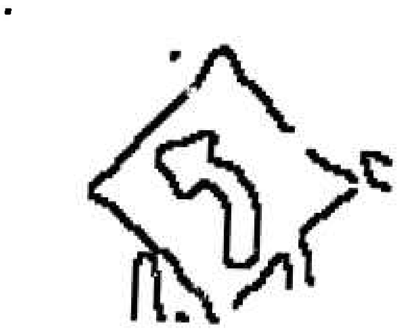


Figure 8b

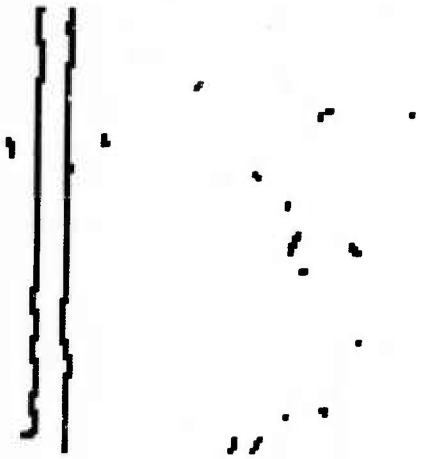


Figure 8c

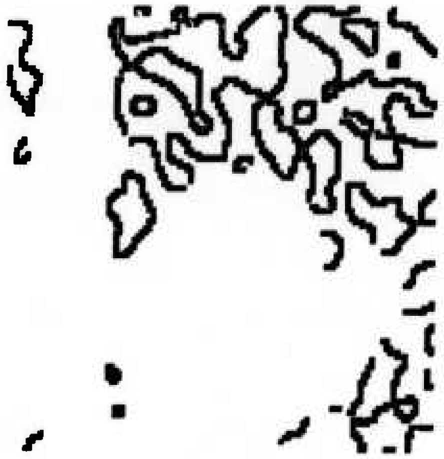


Figure 8d

5.0 DISCUSSION

We now discuss particular aspects of the procedure and then consider several extensions and applications.

5.1 Feature Extraction

Since the procedure's performance does not degrade severely due to the occurrance of poor features, the type of feature extraction used is not critical. Nonetheless, the feature extraction process used here could be extended in many ways. The low-curvature suppression, if it is used, could take into account boundary length along a contour between distinctiveness maxima to determine whether to suppress or generate a feature for further processing. It is also possible to determine points of high curvature along the boundary with out having to walk along the contour by the modifications discussed above in section 1.2 or by using other operators which can directly measure curvature [KIT80].

Another useful extension would be to use information determined from the extraction of the translational axis to isolate false features. This could involve removing those features which have weak matches from the error measure calculation once a translational axis has been determined and re-evaluating. Alternatively, the depth inferences could be used to isolate the positions of potential false features by noting discontinuities in depth along an extracted contour. Extracted features could be removed from the re-evaluation of the error measure if they are at or near such positions. Another type of feature which can affect the evaluation of the error measure are those near an FOE or FOC which is contained in a visible portion of the image. Such features tend to move very small amounts along their image displacement paths and hence require fine interpolation to determine their best matches.

5.2 Properties of the Error Measure

In the experiments, the error measure has a distinct global minimum at the point on the unit sphere corresponding to the correct translational axis. It is expected to have such behavior generally because it is very unlikely that translational axes that are far from the correct position will define image displacement paths that simultaneously allow good matches for many features. Thus competing candidates for the global minimum should not be widely separated.

The error measure is affected by both non-distinctive and false features. Non-distinctive features will match well for many different translational axes. Large numbers of these weak features will flatten the response of the error measure. False features will also distort the error measure since they will often have their best matches with incorrect translational axes.

The effects of these poor features should be compensated by the agreement of good features. Every one of the good features will tend to have a bad match for the incorrect translational axis and their unanimity is expected to overide the lack of discrimination of weak features and the random quality of the matches of false features.

5.3 Utility of the Direction of Translation Sphere

There are significant advantages in defining the error measure with respect to a unit sphere, instead of the potential positions of FOEs and FOCs in the image plane. The sphere is a bounded surface which makes uniform global sampling of the error measure feasible. Additionally, the resolution in the position of the translational axis varies across the surface of the image plane. For example, the FOEs determined by translational axes seperated by very small angles will be seperated by larger and larger distances in the plane as the intersections of the translational axes and the image plane are placed further from the visible image. The effect of this on the error measure, when it is defined over the image plane, is large flat areas for FOEs further from the visible portions of the image. Finally, special criteria must be used to distinguish between FOEs and FOCs if the error measure is defined relative to the image plane. Roughly parallel image displacements could correspond to an FOE off to one side of the image plane or to an FOC off to the opposite side. On the direction of translation sphere, the corresponding translational axes would be close while on the plane they are completely separated.

5.4 Optimization Procedure

The optimization procedure used here is very simple, and, because of the strong unimodality of the error measure and its smoothness, other techniques with more rapid convergence could be used. It is interesting to note, however, that the global component of the optimization performed here is an instance of a generalized Hough Transform [BAL81, O'RO81] in which each feature scales its vote for a particular translational axis by the best match it can find consistent with the translational axis.

6.0 EXTENSIONS AND APPLICATIONS

6.1 Hierarchical Computation

A basic paradigm in computer vision is the use of hierarchical representations and processes. This allows for different magnitudes and scales of image events to be handled uniformly. The consistent agreement among hierarchically organized processes is a basic control strategy for interpretation processes. Additionally, hierarchical processing can produce significant speed-ups wherein results from processing done rapidly at lower resolutions of image information are used to direct and constrain more detailed and extensive processing of higher resolution image information.

The translational motion procedure can be developed in a hierarchical fashion with the primary benefits being increased speed and the ability to deal with large image displacements. This development requires specifying the hierarchical representations of the successive images and the extracted features and how processing at different levels of image resolution are related.

In the initial work described here, images have been represented in the VISIONS image operating cone structure [HAN81]. This consists of a sequence of images I0,I1,I2,...In where the successive sizes of the images are 1x1,2x2,4x4,...,2**n x 2**n. Each pixel in the i-th images, except for the first and last images, has a connected neighboorhood of immediate descendents in the i+1 image and a unique parent in the i-1 image. (One point of confusion: we speak of going up and down the cone and of images having higher and lower resolution. Unfortunately, as we go higher up the cone, image resolution gets lower; and as we go down the cone, image resolution gets higher.) The size and shape of the immediate descendent neighboorhood can be arbitrary. The immediate descendent neighboorhoods of adjacent parallel pixels may or may not overlap.

There are several ways to reduce the resolution of an image and project it up the VISIONS cone [HAN81, BUR82]. These techniques involve smoothing the image with some operator and then sampling at a reduced interval. This can be expressed computationally by expressing the value of a parent pixel to be some average of the pixels in its immediate descendent neighboorhood. The results of reducing image resolution by Gaussian smoothing for the roadsign images is shown in figures 9a-c.

Extracted features can also be represented in the cone structure at different levels of resolution. One approach is to apply the feature extraction process described above for each image at each resulting level of resolution in the cone. Another technique is to extract features in the highest resolution image and project these extracted feature positions up the cone. Thus a feature is positioned at a parent pixel if any of its descendents are at positions where a feature has been extracted. These approaches can interact in interesting ways if the strength of a feature is expressed as a function of the featureness of its ancestors or descendents. Figures 10a-c show the features resulting for the roadsign images at different levels of resolution by projecting extracted feature positions up the cone.

The translational processing can be applied to successive images at any level of resolution for which features have been extracted from the initial image. The basic questions concern how processing at one level effects processing at another level. In particular, how do processing results at a lower level of resolution (higher in the cone!) constrain the processing at higher levels of resolution? At what level in the cone can processing be meaningfully initialized? How do the various parameters involving feature window size, displacement resolution along a flow path, and resolution of the optimization procedure change at different levels of the cone?

For a given pair of images at level i in the cones formed from successive images, the error function is minimized for the set of features determined at level i using projection up the cone from the first image. The determined minimum at level i is then used to constrain the optimization of the error function for the images and feature positions at the next lower level in the cone. In addition to constraints on the position of the error function minimum, processing higher in the cone constrains the evaluation of the potential displacements of extracted features along their displacement paths. For each displacement determined at level i only three positions have to be evaluated at along the displacement paths at level i+1. Thus in processing, not only is the minimum of the error function passed on, but also the displacements of parent features which are then used to constrain the evaluation of the displacements of descendent features at the next lower level.

There are a wide range of possibilities for implementing the error function minimization at the different cone levels. The different resolutions used in the step size of the error function evaluation could be correlated with a particular image level at which processing is being done. That is, as processing proceeds down the cone, the stepsize of the error function evaluation would decrease. Alternatively, a complete search could be done at each level before proceeding further down the cone. Feature size can also change as processing goes down the cone since at higher levels a given window size corresponds to an increased area with respect to the image. At a high level of resolution, features described by small image areas may not be distinctive enough to match well.

In the experiments in figures 11a-c processing was initialized at level 4 (16 x 16 images) by performing the global processing as above. The resulting flow field is shown in figure 11a. The first step of the local processing used a stepsize equal to 0.1 radians and was performed using the images at level 5. The resulting flow field is shown in figure 11b. At level 6, the stepsize was reduced to 0.025 and the local search initialized at the minimum determined by the processing done at level 5. At level 7, the stepsize was reduced to 0.005 and the search was initialized at the minimum determined at level 6. 5x5 windows were used at each level. The procedure converged to the same results as in the experiment above.

An important question concerns the cone level at which to begin processing. One criteria could be the level at which significant changes in image values occur as determined by an average difference value. Another could be the response of the error function. This would involved determining the level at which the error function has a distinct minimum.

Another important question concerns handling features which are on different sides of an occlusion boundary but share a similar ancestor in the feature tree. In this case, the displacement value inherited from the parent may be incorrect for one of the features and the feature should have its potential displacements re-evaluated along it displacement path. A possible criteria to determine the need for re-evaluation of the displacements for a feature is if its match value is ever less than some threshold or is less than the match strength of its parent. It may be sufficient simply to not evaluate such features if they are found and determine their displacements or occlusion after the more certain image displacements have been determined for other image points.

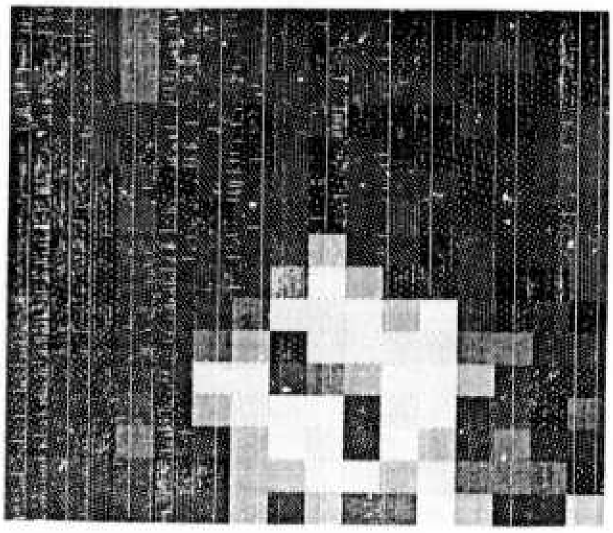


Figure 9a

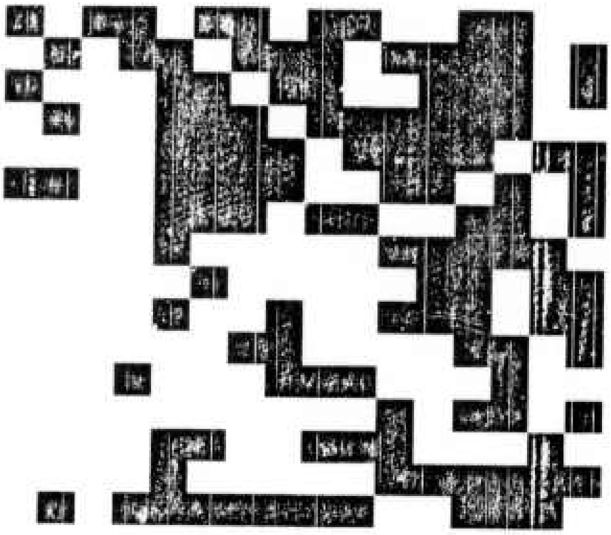


Figure 10a

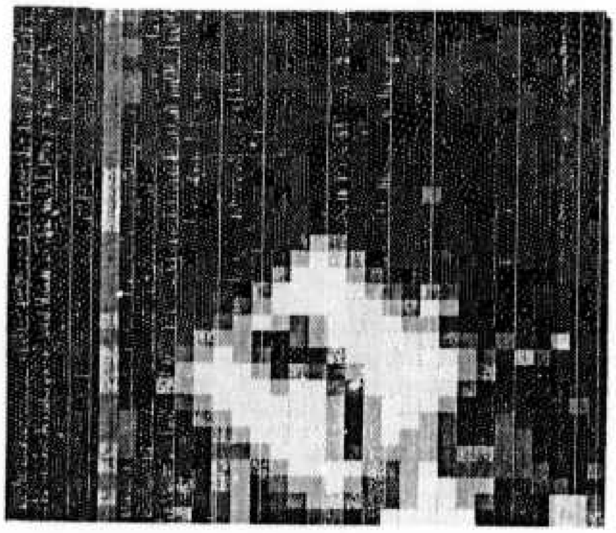


Figure 9b

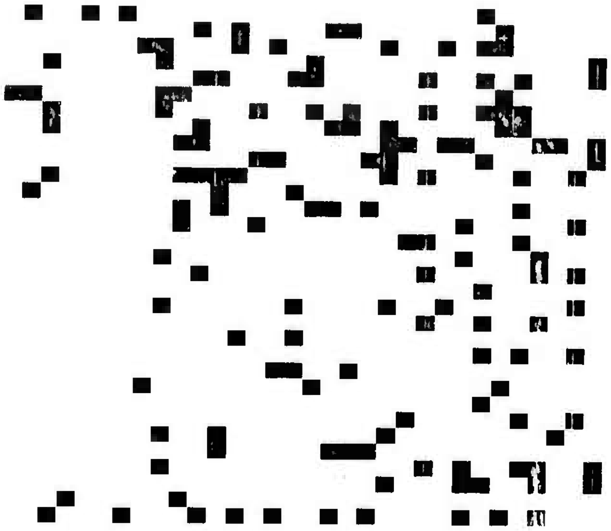


Figure 10b

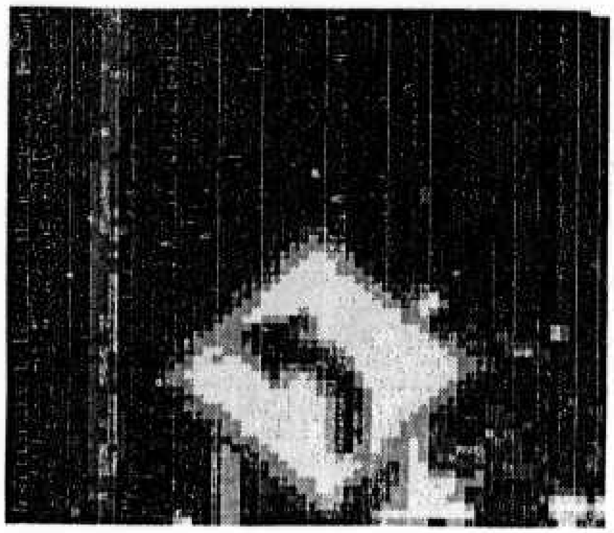


Figure 9c

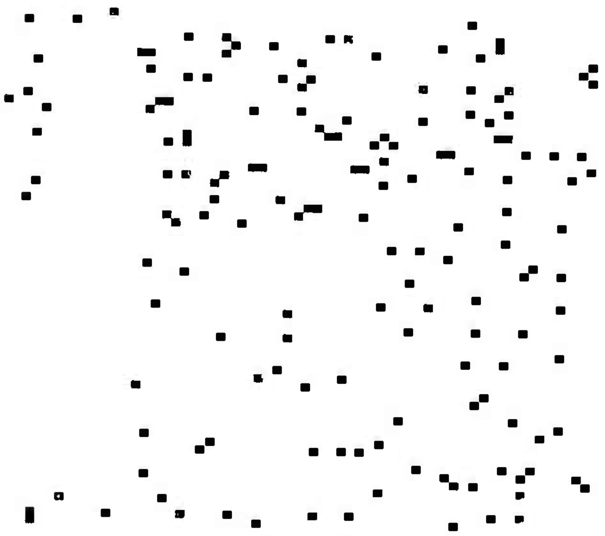


Figure 10c

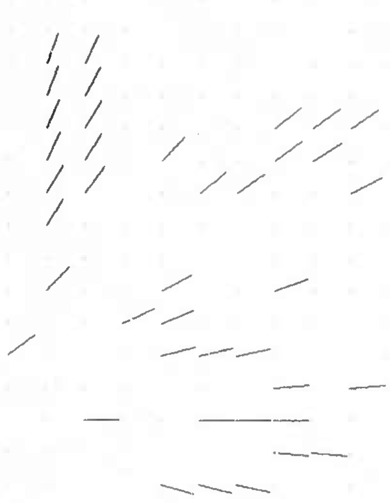


Figure 11a

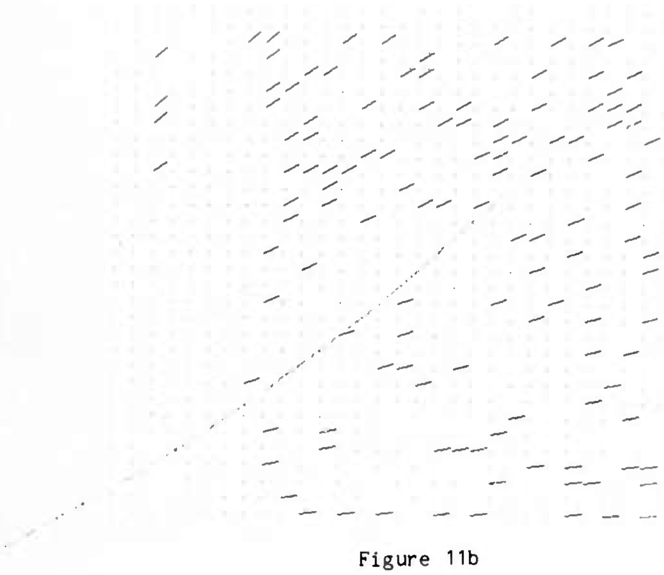


Figure 11b

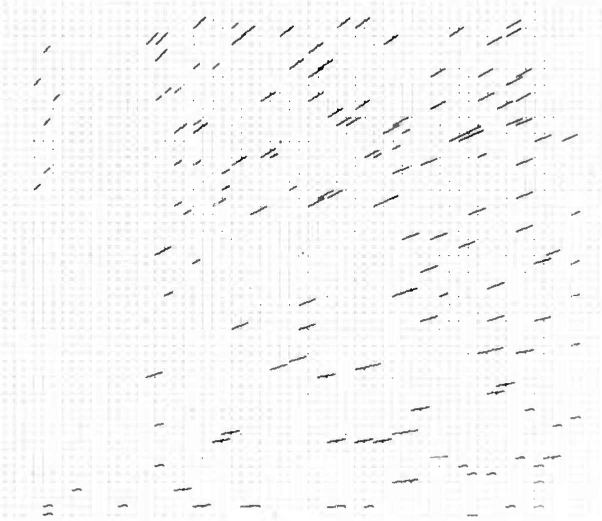


Figure 11c

6.2 Translational Blur Path Extraction

Blur streaks are commonly produced when the shutter mechanism of a camera remains open while the camera is moving relative to a textured surface. The streaks are produced by the successive positions of the image projections of the texture elements. Recent work [HAR80, SHE83] indicates that blur streaks may be a very common motion effect in the human visual system.

For translational camera motion, the blur streaks will correspond to the image displacement paths: straight line segments radiating from a common intersection point. The techniques developed here can be easily extended for the extraction of translation blur paths. First, it is necessary to compute the gradient of the blurred image. The image gradient will be perpendicular to the translational blur paths at image positions which lie along a translational blur path. Thus, the error measure becomes

$$6) \qquad \sum_{i=1}^{n} \text{abs}(\cos\theta_i)$$

where i is an index over image positions and theta(i) is the angle between the image gradient at point i and the translational displacement path corresponding to a particular translational axis. The same evaluation techniques can be used for this error function as above, except that there is no need to distinguish between FOEs and FOCs. Note that in the analysis of translational blur paths, information is lost concerning the direction and magnitude of the displacement of image points over time.

It may be useful to use multiple versions of the same image sequence each formed with a different exposure rate. By substracting the images formed with very short exposure rates (which are basically static images with no dynamic information contained within but edge information corresponding to actual environmental structures) from those with longer exposure rates, it may be possible to suppress edges which are non-blur related in the blurred images. Regardless, the more blurred the images become, the more the static image structure is reduced.

The extraction of translational blur paths is identical to the extraction of vanishing points and lines from static images. This same procedure can be applied, without the initial extraction of edges: indeed, the determination of edges can occur concurrently with the extraction of the vanishing point. We have successfully applied this procedure to the extraction of translational blur paths and the extraction of vanishing points in natural, outdoor images.

6.3 Multiple Independently Moving Objects

The procedure developed here allows for a sensor moving relative to a stationary environment or a single object moving relative to a stationary sensor. A useful extension would allow for multiple, independently moving objects while maintaining the ability to determine image displacements concurrently with the direction of translation. There are at least three techniques which would make this possible. One is to utilize generalized hough transform techniques for decomposing the responses in a histogram into the corresponding image structures or segments. The others utilize the limited image areas over which the procedure can successfully function.

As pointed out in the discussion above, the global sampling of the error function is an instance of a generalized Hough transform. Each feature is scaling its vote against a particular translational axis by the extent of feature mismatch it has along an image displacement path determined by the translational axis. Without changing anything, and to be consistent with other developments, instead of using an error measure, we could use an optimization measure by which each feature scales its vote for a particular translational axis by the extent of match it has consistent with the axis. The problem then becomes a typical one for generalized Hough transforms: how to associate labels corresponding to the resulting peaks in the histogram with image points or features. The general form of this processing is to find the greatest response in the hough transform, associate a label with it, and then associate this label with, in this case, image features which match above some threshold (corresponding to strength of match) along the path determined by the axis. The resulting set of features are then removed and a new histogram is produced (or rehistogramming). The peak in this new histogram is determined, a new label associated with it and mapped onto the corresponding image features. This process is repeated until there are no more distinct peaks in the resulting hough transforms or all image features are labeled.

Unfortunately, this procedure will have difficulties with weak or homogeneous feature points which have strong matches consistent with several distinct translational axes. Thus, when rehistogramming occurrs it is necessary to establish which image features, which are already labeled, are consistent with the newly extracted peak. This is costly and could be quite messy. An alternative, is to proceed in the conventional manner and determine a set of labels corresponding to translational axes for which there is evidence. Each feature is then labeled with each translational axis from this set with which it is consistent. Note that a given feature could have several labels. A unique consistent labelling is then obtained by using other information: segmentation-grouping using other image attributes, depth consistency with neighboors, and common magnitude of image displacements. Additionally, this disambiguation can occur over several successive images.

Two basic questions to be addressed in this use of Hough techniques are what is the required density of translational axes in the transform and what is the minimal match threshold.

An alternative approach is to break the image into subparts and then locally apply the procedure to associate a translational axis with each subpart. In one scheme, this would be done using regular image areas (as in a grid). In another scheme, the subparts are determined by some segmentation procedure and the translational axis is determined from image features within or lying along the boundary of the extracted segments. Segments for which the error function response is indistinct are resegmented or their features are associated with the translational axes determined for adjacent image subparts.

6.4 Local Translational Decomposition

The technique for translational motion processing can be extended to less restricted forms of sensor motion by applying the procedure to small areas across an image surface over a sequence of images. This approximates more general motions as consisting locally of environmental translations and interprets local image motion as resulting from environmental translations. The feasibility of this is based upon experiments showing that the direction of translation can be extracted with reasonable precision using small image areas containing a few features. The resulting description associates with a set of image points (or small image areas) the approximated direction of motion of the corresponding environmental points (or small environmental surface area). As a low level representation of environmental motion, this can considerably simplify the recovery of the sensor motion parameters [LAW82]. It can also provide qualitative information concerning the rough motion characteristics of objects in a scene.

6.5 Other Cases of Restricted Motion

The procedure developed here is applicable to other cases of unknown but restricted camera motions for which it is computationally feasible to search directly through a subspace of the camera motion parameters. Two particular cases are pure sensor rotation and motion constrained to a known plane.

With pure sensor rotation, the unknown camera parameters are constrained to R1(t), R2(t), and R3(t). In this case, the error measure is defined with respect to a direction of rotation sphere where each point corresponds to an axis of rotation. For each rotational axis, the extent of displacement for image features is determined by different values of R3(t). There is the additional constraint in this case that the displacements of all features must correspond to the same value of R3(t). Thus the variance of the determined angular displacements can be incorporated into the error measure.

For motion constrained to a known plane, the rotational axis is known to be perpendicular to the plane and the translational axis is constrained to lie in it. Thus, only R3(t) and one translational parameter can vary and the error measure can be computed with repect to these two parameters. The global sampling in this case amounts to evaluating a set of translational axes for each of a set of potential rotations.

6.6 Hybrid Sensor Systems

Translational processing is sufficient for vision-based navigation in a stationary environment if the orientation of the optic sensor can be fixed relative to the environment over time. In this case, sensor motion amounts to a sequence of translations in possibly different directions over time.

A difficulty with such a stabilized retina is that much of the environment would not be observable. This can be corrected by using a set of such stabilized retinas arranged to yield a complete view of space. There would then be no need to rotate the sensor to view a particular environmental point. A possible arrangement of retinal surfaces is a cubical one. One of the retinal planes will always contain an FOE and another will always contain an FOC (unless the direction of motion is right on an edge of the cube and the focal length has not been properly adjusted). There will also be several independent estimates of the direction of translation which can be integrated.

Alternatively, if the sensor can not be stabilized, there are devices which can at least determine the rotational parameters of sensor motion. The rotational effects can then be removed from successive images, reducing them to translational sequences which can be processed by the techniques here. A particular technology which is very attractive for this use is Fiber Optic Rotation Sensors [EZE82] which are expected to be the low-cost gyroscope of the near future. These devices are small, cheap, and precise. There are currently slow drift problems, but we would be concerned with measurements of rotation over very short periods. Additionally, when coupled with an image processing system, such long term drifts could be recognized and accounted for by noting the position of specified landmarks.

7.0 CONCLUSIONS

This work demonstrates a simple and robust procedure for determining the direction of environmental motion and image displacements in real-world image sequences produced by translation. It is not dependent on an initial matching process prior to the inference of camera motion. Instead, features are extracted from an initial image and their displacements are determined concurrently with the inference of direction of sensor motion. Thus complications in matching that arise from an individual feature being extracted in one image and not in the next are reduced. The process is also relatively insensitive to weak and false features. It can use a small number of features positioned across an image surface or a small number of features from a limited area of the image. It has been successfully applied to image sequences produced by a car translating down a road, by a camera attached to a robot manipulator in an industrial environment, and to artificially generated sequences.

We further considered and demonstrated several extensions and applications for such things as independently moving objects, translational blur streaks, other cases of restricted motion, computation in a hierarchical structure, and potential incorporation into hybrid sensor systems for autonomous navigation.

REFERENCES

[BAL81] Ballard, D.H., "Parameter Networks: Towards a Theory of Low-Level Vision," Proc. of 7th IJCAI, Vancouver, British Columbia, August 1981, pp. 1068-1078.

[BUR82] Burt, P. J., "The Pyramid as a Structure for Efficient Computation", Image Processing Laboratory Technical Report IPL-TR-038, Electrical and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, New York 12181, 1982.

[DUD73] Duda, R.O. and Hart, P.E., Pattern Classification and Scene Analysis, New York. Wiley 1973.

[EZE82] Ezekiel, S. and Arditty, H.J., Fiber-Optic Rotation Sensors and Related Technologies, New York, Springer-Verlag, 1982.

[HAN81] Hanson, A. R. and Riseman, E. M., "Processing Cones: A Computational Structure for Image Analysis", in Structured Computer Vision, S. Tanimoto and A. Klinger (Editors), Academic Press, 1980.

[HAR80] Harrington, T. L., Harrington, M. K., Wilkins, C. A. and Koh, Y. O. "Visual Orientation by Motion-Produced Blur Patterns: Detection of Divergence", Perception and Psychophysics, vol. 28, pp. 293-305, 1980.

[HIL80] Hildreth, E.C., "Implementation of a Theory of Edge Detection," MIT AI Technical Report AI-TR-579, MIT, Cambridge, MA, 1980.

[KIT80] Kitchen, L. and Rosenfeld, A. "Gray-Level Corner Detection", Tr-887, Computer Vision Laboratory, Computer Science Center, University of Maryland, College Park, MD 20742, April, 1980.

[LAW82] Lawton, D. T., "Motion Analysis via Local Translational Processing", IEEE Workshop on Computer Vision: Representation and Control, Rindge, New Hampshire, August, 1982.

[LAW83] Lawton, D. T., and Weems, C. "Incorporating Content Addressable Array Processors into Computer Vision Systems", in preparation.

[LEE80] Lee, D. N., "The Optic Flow Field: The Foundation of Vision," Philosophical Trans. Royal Soc. London, Volume B, Number 290, 1980, pp. 169-179.

[MAR80] Marr, D. and Hildreth, E., "Theory of Edge Detection," Proc. Royal Soc. London, Volume B, 1980, pp. 187-217.

[MOR80] Moravec, H. P., "Rover Visual Obstacle Avoidance," Robotics Institute, Carnegie-Mellon University.

[MOR77] Moravec, H. P., "Towards Automatic Visual Obstacle Avoidance," Proceedings of the 5th IJCAI, MIT, Cambridge, MA, 1977, p. 584.

[O'RO81] O'Rourke, J. "Motion Detection Using Hough Techniques", Proceedings of PRIP 1981, pp. 82-87.

[PRA81] Prazdny, K., "Determining the Instantaneous Direction of Motion from Optical Flow Generated by a Curvilinearly Moving Observer", Proc. of the Pattern Recognition and Image Processing Conference, Dallas, Texas, August 1981, pp. 109-114.

[SHE83] Shepard, R. N. and Zare, S. L., "Path-Guided Apparent Motion", Science, vol. 220, pp. 632-634, 1983.

THREE-DIMENSIONAL SHAPE FROM LINE DRAWINGS

Stephen T. Barnard and Alex P. Pentland

SRI International, 333 Ravenswood Ave., Menlo Park, California 94025

ABSTRACT

The problem of interpreting the shape of a three-dimensional space curve from its two-dimensional perspective image contour is considered. Observation of human perception indicates that a good strategy is to segment the image contour in such a way as to obtain approximately planar segments. The orientation of the osculating plane (the plane in which the space curve lies) can then be estimated for these segments, and the three-dimensional shape recovered. The assumption of spatial isotropy is used to derive the theoretical results needed to formulate such an estimation strategy. The resulting estimation strategy allows a single three-dimensional structure (up to a single Necker reversal) to be assigned to any smooth image contour. An implementation is described and shown to produce an interpretation that is quite similar to the analytically correct one in the case of a helix, even though a helix has substantial torsion. The general applicability of the algorithm is discussed.

I Introduction

Much recent vision research has emphasized the importance of image contour for shape interpretation [1,2,3,4,5,6,7]. Tenenbaum and Barrow [1] argue that image contour, for example, is dominant over shape from shading. Pentland [8] has presented examples in which the addition of a contour substantially improved the interpretation of a shaded surface. It seems that contour is one of the strongest sources of information for shape perception.

One source of evidence of the strength of contour information is line drawings. When we examine a line drawing, our perception of the three-dimensional shape implied by such a drawing is nearly always clear and unambiguous. How can we account for this, given that purely geometrical constraints admit of an infinite number of valid interpretations?

A. An Observation About Human Perception

When we observe line drawings such as those in Figure 1 (a), we have a clear perception of a non-planar three-dimensional structure. Notice that if we were to segment each of these drawings at the circled points, each of the resulting segments would have the same shape as they did when they were still hooked together *and would be approximately planar*, as is shown in Figure 1 (b). Thus, for these line drawings the problem of recovering the three-dimensional structure can be reduced to the problems of (1) segmenting the curve into perceptually planar segments, and (2) finding the plane that contains each of the curve segments (the *osculating plane*) [9]. Once we know the orientation of the plane which contains a curve segment we can then easily determine its three-dimensional shape.

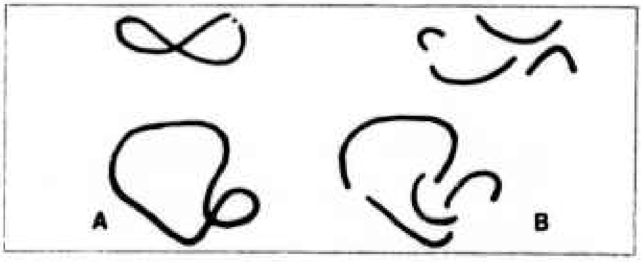


Figure 1. (a) Some Line Drawings, (b) Their Planar Subregions.

If we "by hand" try to segment image contours into planar regions, we find that the strategy can be successfully applied to a surprisingly large number of naturally-occurring image contours. For some contours, however, it is not obvious how well this strategy will work, primarily because there are no points which segment the space curve into planar regions. An example of such a curve is the helix shown in Figure 2 (a). Nonetheless, it may still be possible to obtain a good approximation of the three-dimensional structure of such a curve using this strategy.

B. A Strategy For Recovering Three-Dimensional Shape

This observation about human perception leads to the following processing strategy:

(1) Segment the image contour in such a way that each segment is likely to comprise a projection of a planar segment of the space curve.

(2) Calculate the planes implied by the segments from (1).

(3) Assemble the results of (2) into an estimate of the shape of the entire space curve.

The specific criteria for the initial segmentation are not dealt with here. It is clear, however, that the image contour should be segmented at singular points of curvature (maxima, minima, and inflection points). Hoffman and Richards [10] have presented a theory of curve segmentation that addresses this issue. Our approach will be to temporarily ignore the segmentation problem and to simply estimate the orientation of parts of the space curve from many local parts of the image contour. If valid results are forthcoming with this approach the method can only be improved with more elaborate segmentation.

C. Modeling the Space Curve

We shall model a space curve in the conventional way, as a three-dimensional vector function $\mathbf{x}(s)$ of one parameter s which is assumed to be a natural parameter, i.e., $|d\mathbf{x}(s)/ds| = 1$. The shape of such a curve is completely determined by two properties that are scalar functions of s: curvature, $\kappa(s)$, and torsion, $\tau(s)$ [9]. Curvature is always nonnegative; only straight lines and inflection points have zero curvature. Torsion may be intuitively defined as the amount of "twist" in the curve at a point s. Another way to visualize torsion is as the degree to which the osculating plane (the plane which contains the curve) is changing. Only planar

* The research reported herein was supported by the Defense Advanced Research Projects Agency under Contract No. MDA 903-83-C-0027; this contract is monitored by the U. S. Army Engineer Topographic Laboratory. Approved for public release, distribution unlimited.

curves have zero torsion everywhere. Unlike curvature, torsion may be either negative or positive.

The presence of torsion is not directly evident in the image. It simply results in more or less foreshortening as the osculating plane of the contour varies. The effects of torsion, therefore, can be exactly mimicked by changes in curvature, and vice versa.

II Theory of Contour Interpretation

Not all three-dimensional interpretations of an image contour are equally likely. If we assume that spatial isotropy holds, then we know that viewer position is independent of the shape of the curve — which allows us to make a reasonable guess about the latter's three-dimensional shape [8]. The first step towards a guess at the space curve's shape is the following proposition:

Proposition (Zero Torsion). The maximum-likelihood estimate of the torsion of the space curve is zero (i.e., no "twisting" of the curve).

This proposition follows because the assumption of spatial isotropy implies that the viewer's position and the shape of the space curve are mutually independent. Thus, not only is it unlikely that significant features of the curve will be hidden from view by coincidental alignment of the viewer and the curve, but, conversely, it is likely that the viewed scene will not change much with small changes in viewing position.* The appearance of a curve with substantial torsion** will change considerably with small changes in viewer position; if we assume spatial isotropy, therefore, we must expect that the torsion of the curve will be small.

Furthermore, given that spatial isotropy implies that the viewer position and the shape of the curve are mutually independent, the torsion of the curve must then also be independent of viewer position. Consequently, the torsion of the curve is as likely to be positive as negative, and thus the mean value (and maximum-likelihood estimate) for the magnitude of the torsion is zero† . The probability that the torsion is small implies this estimate will generally be a good one.

A. Estimation With The Assumption Of Zero Torsion

Even if we assume that torsion is zero (i.e., the space curve is planar), there is still a two-parameter set of space curves that could have generated that imaged contour. The two parameters correspond to the two degrees of freedom of the osculating plane.

Assume that we are given a small portion of an imaged contour, and asked to estimate the three-dimensional shape of the space curve which generated that image. If we measure the position and curvature at three points on the imaged contour, then we can uniquely define an elliptical arc that fits the image data. By the previous proposition, this elliptical arc is most likely caused by a space curve that is either an arc of a circle or of an ellipse, as those are the two planar (zero torsion) shapes which can project to an ellipse†† .

Previous research ([2], [12]) has shown that the maximum-likelihood estimate of the space curve's shape is given by the following proposition (see also [2]):

Proposition (Planar Interpretation). Given an elliptical segment of an image contour and that the space curve is planar, the maximum likelihood estimate of the space curve's three-dimensional shape is a segment of a circle.

Barnard [12] has constructed a maximum entropy estimator that implements this proposition for perspective images and that is tolerant of digitization noise. Operating under the assumption that the space curve has zero torsion, it chooses the orientation that maximizes the entropy of backprojected image contour curvature measurements. That is, curvature is first measured at several points in the image contour, then the curvatures of hypothetical planar space curves of essentially all orientations are computed by backprojection, and, finally, the orientation that leads to the space curve of most uniform curvature (in the sense of maximum entropy) is selected. In general, three image contour curvature measurements are sufficient for an unambiguous maximum-entropy interpretation (up to a Necker reversal).

III Three-Dimensional Estimation

Now let us return to the general problem of estimating the shape of the space curve, given a smooth imaged contour. Let us first take three curvature measurements along the imaged contour. These three measurements define an ellipse. As just described, this leads to a circular interpretation of the space curve. Now suppose that we have additional image contour curvature measurements. There are, then, two cases to consider:

First case: the new points fit on the same ellipse. In the first case we have quite strong evidence of the space curve's shape. For, if the osculating plane were changing, the curvature would have to be changing also — and in just such a manner as to exactly cancel (in the image) the effect of the changing osculating plane. Similarly, if the curvature of the space curve were changing, the osculating plane would have to change just exactly enough to cancel the effect of the changing curvature. As such a "conspiracy" to cancel the visible effects of change is unlikely (a direct violation of general position), we must conclude that there was neither torsion nor change in curvature, and, thus, there is a great (in fact, maximum) likelihood that the new image curvature measurements result from the same circular space curve defined by the first three measurements.

Second case: the new points don't fit on the same ellipse. What if the additional measurements lie off the ellipse defined by the first three measurements? Then we can be certain that either the curvature or the osculating plane (or both) of the space curve has changed. This new point is, therefore, a possible place to segment the curve. What we must do when we encounter such a point is advance along the image contour until we are completely past the point, and obtain a new estimate of the space curve's osculating plane. If the new osculating plane has the same orientation as the previous osculating plane, then we have evidence that the space curve continues to be planar, and we should not segment the curve. If, however, we obtain a different orientation for the osculating plane, then we should segment the space curve and begin a new planar segment of the curve.

As any smooth image contour may be closely approximated by portions of ellipses and straight lines* , this interpretation strategy will yield a single interpretation for the three-

*This is often referred to as the assumption of general position. Thus, spatial isotropy implies general viewing position.

**As a function of position on the image contour rather than as a function of s

†Note that at places where the curvature is zero — straight segments and inflection points — the torsion is not defined and may arbitrarily be taken to be zero. That is, the osculating plane may be changed freely at these points without affecting the shape of the space curve.

††This is true of both perspective and orthographic projection, however, we will deal exclusively with the more general case of perspective foreshortening.

*Only the third and higher derivatives of the imaged contour that will fail to be exactly matched. People, it should be noted, are very poor observers of changes in the third derivatives of an image contour.

dimensional shape of the space curve (up to Necker reversals). Further, this interpretation will be the most likely interpretation on a point-by-point basis. It should be noted that the first two steps of this estimation strategy are similar to the strategy proposed in [1].

IV An Example

The interpretation strategy has been implemented and applied to a synthetic image of a helical space curve. The helix example is a good test because a helix has significant torsion everywhere, thus, distinguished segmentation points do not exist and it is not clear what the estimation strategy will do. If we can recover the helical shape of the space curve with some accuracy, we shall have demonstrated that the estimation strategy can perform even when no good segmentation is available.

Figure 2 (a) shows a perspective image of a helix. Figure 2 (b) shows a plot of the spherical indicatrix of the helix. The spherical indicatrix is a plot of the orientation of the osculating plane of the space curve. The axes in this plot corresponds to the azimuth and elevation of the osculating plane. As mentioned previously, knowledge of the orientation (azimuth and elevation) of the osculating plane at each point, together with the imaged contour, uniquely determines the shape of the space curve. Thus, the spherical indicatrix is a method of displaying the three-dimensional shape of the space curve. Figure 2 (c) shows the spherical indicatrix estimated for the contour in (a). When this is compared with the actual indicatrix shown in (b), it is evident that the three-dimensional shape of the space curve has been fairly accurately recovered.

Summary. We have developed a theory for assigning a three-dimensional interpretation to any smooth image contour. The theory has been implemented and is undergoing evaluation, which may lead to further development. The results reported above indicate that the estimation strategy performs reasonably well even for cases such as a helix, where the presence of substantial torsion might have led one to expect poor performance.

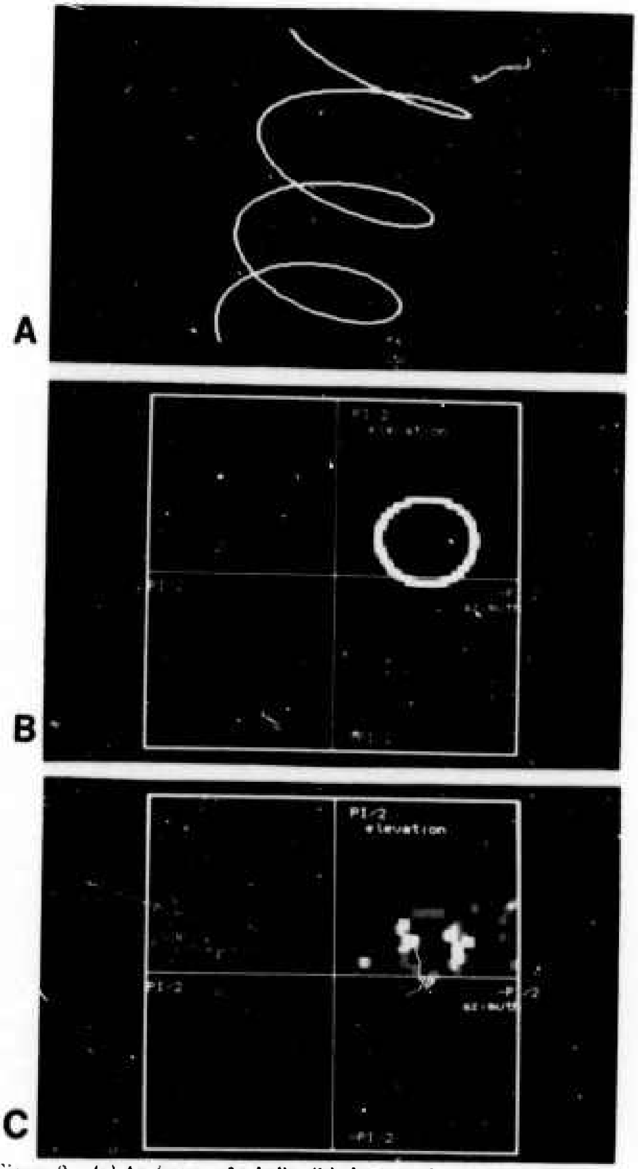


Figure 2. (a) An image of a helix, (b) the actual spherical indicatrix, (c) the recovered spherical indicatrix.

REFERENCES

[1] H. G. Barrow and J. M. Tenenbaum, Interpreting Line Drawings as Three-Dimensional Surfaces,*Artificial Intelligence* **17** (1981), 1-47.

[2] A. Witkin, Recovering Surface Shape And Orientation From Texture,*Artificial Intelligence* **17** (1981), 17-47.

[3] J. Kender, Shape From Texture,*Ph. D. Thesis, Computer Science Department, Carnegie-Mellon University* (1980).

[4] T. Kanade, Recovery of the 3-D Shape of an Object from a Single View,*Artificial Intelligence* **17** (1981), 409-460.

[5] K. A. Stevens, The Visual Interpretation of Surface Contours,*Artificial Intelligence* **17** (1981), 47-73.

[6] M. Brady and A. Yuille, An extremum principle for shape from contour,*In this proceedings* .

[7] D. G. Lowe and T. O. Binford, The Interpretation of Three-Dimensional Structure from Image Curves,*Proceedings of the 7th IJCAI, Aug. 24-28, University of British Columbia, Vancouver, B.C., Canada.* **2**, 613-618.

[8] A. Pentland, Local Inference of Shape: Computation From Local Features,*Ph.D. Thesis, Psychology Department, Massachusetts Institute of Technology* (1982).

[9] M. M. Lipshutz, *Differential Geometry*, McGraw-Hill, New York, New York, 1969.

[10] D. D. Hoffman and W. A. Richards, Representing Smooth Plane Curves for Recognition: Implications for Figure-Ground Reversal,*Proceedings of the AAAI, Aug. 18-20, 1982, Carnegie-Mellon University, Pittsburg, Pennsylvania* , 5-8.

[11] A. Witkin, Shape From Contour,*Ph.D. Thesis, Psychology Department, Massachusetts Institute of Technology* (1982).

[12] S. Barnard, Interpreting Perspective Images,*SRI Artificial Intelligence Center Technical Note 271* to appear in *Artificial Intelligence* (1982).

Dup

RECOVERING 2-D MOTION PARAMETERS IN SCENES CONTAINING MULTIPLE MOVING OBJECTS

Gilad Adiv

Computer and Information Science Department
University of Massachusetts
Amherst, MA 01003

Abstract

A method for extracting the motion parameters of several independently moving objects from displacement field information is described. The method is based on a generalized Hough transform technique. Some of the problems of this technique are addressed and appropriate solutions are proposed. A modified multipass Hough transform approach has been implemented, where in each pass windows are located around objects and the transform is applied only to the displacement vectors contained in these windows. The windows are determined by the degree to which the displacement field is locally inconsistent with previously found motion transformations. Thus, the sensitivity of the Hough transform to local events is increased and the motion parameters of small objects can be detected even in a noisy displacement field. We also use a multi-resolution scheme in both the image plane and the parameter space and thus reduce the computational cost of the technique. The method is demonstrated by experiments based on artificial images with four parameters of 2-D motion: rotation, expansion and translation in both axes.

1. Introduction

A time-varying scene may contain several independently moving objects with unknown location, shape and 3-D structure. The interpretation of such a scene includes the computation of the motion parameters of the camera and each moving object. This information is useful in areas such as robotics and navigation. It could also be used as an intermediate stage for achieving the tasks of object-surround separation and structure determination.

Our approach for recovering the motion parameters is based on two phases. First, we compute a displacement field, composed of vectors describing the displacement of image elements from one image to the next (see section 2). In this paper we assume a dense displacement field, but the second phase is basically independent of this assumption. Each displacement vector is assigned a weight representing its reliability .

In the second phase the displacement field is interpreted and the motion parameters are recovered. This phase, which is the main concern of the paper, is based on the generalized Hough transform technique [BAL81a]. In this technique the motion parameters are represented by a discrete multi-dimensional parameter space where each dimension corresponds to one of the parameters. Each point in this space uniquely characterizes a motion transformation, defined by the corresponding parameter values. A displacement vector "votes" for a point in the space if the corresponding transformation is consistent with this vector. The points receiving the most votes are likely to represent the motion parameters of different objects.

There are a few techniques described in the literature which use the Hough transform for dealing with scenes containing several moving objects. Fennema and Thompson [FEN79] compute spatial and temporal gradients of the image. A Hough transform technique is used to detect velocities which are consistent with a significant portion of the gradient field. A multipass approach is used: first the most prominent peak in the Hough transform is found and thus the velocity of the largest object is recovered. Then the image points which are consistent with this velocity are removed and a new peak is looked for. The process is repeated until no further objects are found. This system is restricted to translation. It also has problems in recognizing significant peaks [THO81].

Ballard and Kimball [BAL81b] consider the case of general 3-D motion of rigid objects, but assume knowledge of depth information. A Hough transform technique for computing the motion parameters from 3-D optic flow is implemented. The simulation, as described in their report, assumes only one moving object, but it is argued that a multipass approach would handle the case of several moving objects.

Jayaramurthy and Jain [JAY82] describe an implementation of the Hough transform technique for computing motion parameters directly from the intensity information. Several moving objects are allowed, but a stationary background and translational motion are assumed.

One of the well known advantages of the Hough technique is its relative insensitivity to noise and partially incorrect or occluded data. Another advantage is its ability to detect consistency in the image. In our case it can group together displacement vectors which satisfy the same motion parameters and presumably belong to one object.

On the other hand, the Hough technique has a few disadvantages. It is insensitive to spatial relations in the displacement field. Thus, a group of non-adjacent elements, which incidently vote for the same motion transformation, may be considered as representing one object, whereas the motion parameters of a small object may be difficult to detect. The technique also has high computational cost. Fine resolution in the parameter space, which is related to the accuracy of the final results, requires large amounts of memory and computation time.

This paper addresses these problems. A few ideas are examined in a restricted case of 2-D motion with four parameters (rotation, expansion and translation in both axes). An analysis of reliability and efficiency considerations is presented (section 3.2) and new solutions are proposed (section 4). A modified multipass Hough transform approach has been implemented, where in each pass windows are located around objects and the transform is applied only to the displacement vectors contained in these windows. The windows are determined by the degree to which the displacement field is locally inconsistent with previously found motion transformations. Thus, the sensitivity of the Hough transform to local events is increased and the motion parameters of small objects can be detected even in a noisy displacement field. We also use a multi-resolution scheme in both the image plane and the parameter space and thus reduce the computational cost of the technique. These ideas are demonstrated by experiments based on artificial images (section 5).

2. Computing a Displacement Field and a Weight Plane

In the first phase of the algorithm we compute a displacement field from two sampled images. These images contain several objects which are moving independently. The background is considered as one of the objects. The motion of each object is composed of rotation, expansion and translation. It can be represented by the following affine transformation:

$$(2.1) \qquad i' = (1+expan)[\cos(rot)\, i-\sin(rot)\, j]+tr_1$$

$$(2.2) \qquad j' = (1+expan)[\sin(rot)\, i+\cos(rot)\, j]+tr_2$$

where (i,j) is a pixel in the first image, (i',j') is the corresponding pixel in the second image and rot, expan, tr_1 and tr_2 are the motion parameter values.

The displacement field can be described by $\{(D_1(i,j), D_2(i,j))\}$ where $(D_1(i,j), D_2(i,j))$ represents the displacement vector at the (i,j) pixel in the first image. We compute it by using the Horn and Schunck technique [HOR80] (however, the second phase of our algorithm is almost independent of this specific choice). In order to use this tehnique we assume a small displacement at each pixel and absence of illumination effects. It starts by calculating, at each pixel, the spatial gradient (E_1, E_2) and the temporal derivative E_t. The assumption that the brightness of a particular point in the scene is constant over time provides the following constraint:

$$(2.3) \qquad E_1D_1+E_2D_2+E_t = 0$$

The assumption of the smoothness of the displacement field provides another constraint.

An error function can represent, for a given displacement field, the degree of departure from these constraints. The technique is based on iteratively minimizing this function. Ideally, the resulting field (D_1, D_2) should satisfy the following equations, derived from equations (2.1) and (2.2):

$$(2.4) \qquad i+D_1(i,j) = (1+expan)[\cos(rot)\, i-\sin(rot)\, j]+tr_1$$

$$(2.5) \qquad j+D_2(i,j) = (1+expan)[\sin(rot)\, i+\cos(rot)\, j]+tr_2$$

where rot, expan, tr_1 and tr_2 are the motion parameter values in the (i,j) pixel.

Figure 1 shows two pairs of artificial images which contain several independently moving objects. The motion parameters of each object are specified in tables 5.1 and 5.2. Figure 2 shows the result of applying the Horn and Schunck technique to these images.

The smoothness constraint is violated at the boundaries of independently moving objects. Therefore, the computed displacement values in these areas are incorrect. Fortunately, these areas can be detected by using the error function which represents the departure from the constraints. High values of the error function indicate that the constraints are not satisfied and the computed displacement values are unreliable.

For each displacement vector we compute an associated weight such that high reliability (low value of the error function) is represented by a value close to 1 and low reliability by a value close to 0. An appropriate relation between the error function, erf(i,j), and the weight, W(i,j), can be obtained by the function

$$(2.6) \qquad W(i,j) = e^{-erf(i,j)/k}$$

The parameter k was experimentally determined as 0.07. However, this value need to be decreased with noisier data. Figure 3 shows the weight planes computed for the displacement fields in figure 2. When the Hough transform is computed later the influence ('voting' power) of each displacement vector will be proportional to its associated weight.

3. The Generalized Hough Transform Technique

3.1 General Description

The motion parameters can be represented by a 4-dimensional parameter space where each dimension corresponds to one of the motion parameters: rotation (rot), expansion or contraction (expan), vertical translation (tr_1) and horizontal translation (tr_2). Each point in this space uniquely characterizes a motion transformation in the image.

We say that a displacement vector $(D_1(i,j), D_2(i,j))$ is consistent with a point $(rot,expan,tr_1,tr_2)$ in the parameter space if it satisfies equations (2.4) and (2.5). Let us define a subset B(i,j) of the parameter space as the set of all the points in this space which are consistent with

$(D_1(i,j), D_2(i,j))$. Using the definition in [BAL81a], the Hough transform is the following function, defined on the parameter space:

$$(3.1) \quad H(rot,expan,tr_1,tr_2) = \sum_{(rot,expan,tr_1,tr_2) \in B(i,j)} W(i,j)$$

i.e., $H(rot,expan,tr_1,tr_2)$ is the sum of the associated weights of all the displacement vectors which are consistent with the point $(rot,expan,tr_1,tr_2)$. High values of the Hough transform represent motion transformations which are consistent with a significant portion of the vectors. The use of the weight function W should prevent unrcliable values of displacement vectors from creating false peaks.

In practice, we assume a limited velocity of objects, i.e. minimal and maximal values for each parameter. The corresponding intervals are quantized and thus each parameter is represented by a discrete set of values. The parameter space is the cartesian product of these sets.

For each displacement vector $(D_1(i,j), D_2(i,j))$ and each pair (rot,expan) of rotation and expansion, there exists exactly one pair of translations (tr_1^*,tr_2^*) which satisfies equations (2.4) and (2.5). If tr_1^* and tr_2^* are within the limits of the respective dimensions of the parameter space, then we can find exactly one pair (tr_1,tr_2) such that tr_1 and tr_2 are sampled values and

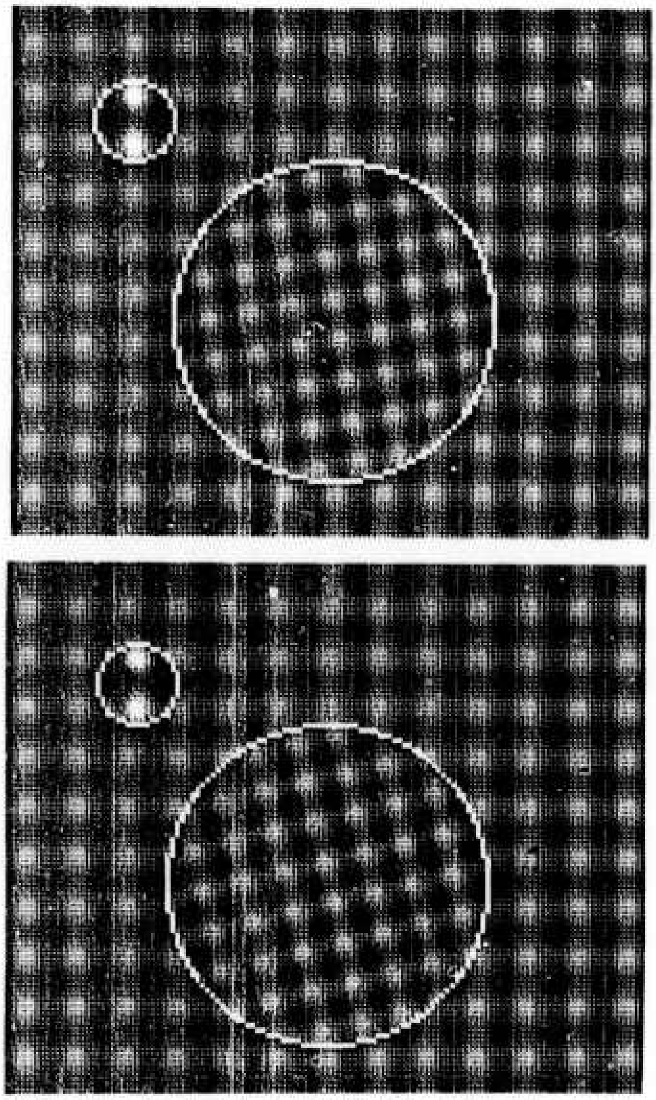


Figure 1.a: Intensity images used in the first experiment (the white lines only emphasize the contours of the objects and are not part of the images);
- object A is the background,
- object B is the large circle in the center of the image,
- object C is the small circle in the upper left corner.

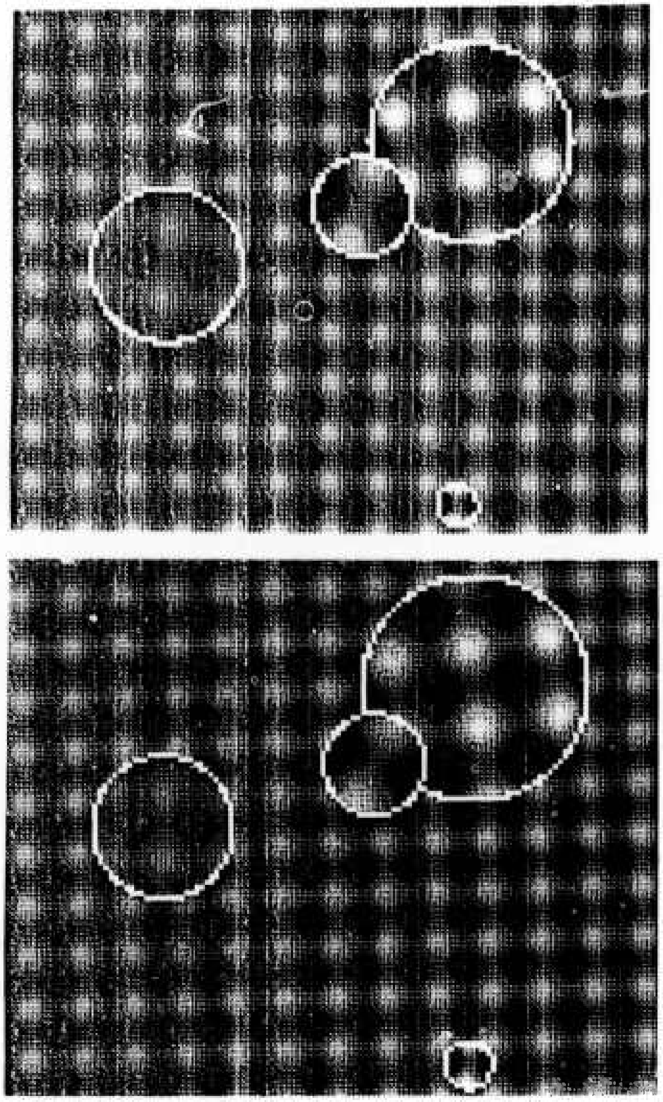


Figure 1.b: Intensity images used in the second experiment;
- object A is the background,
- object B is the circle in the upper right corner,
- object C is the circle which partially occludes object B,
- object D is the circle in the left part of the image,
- object E is the small circle in the lower part.

(3.2) $\quad tr_1\text{-}res/2 \le tr_1^* < tr_1+res/2$

(3.3) $\quad tr_2\text{-}res/2 \le tr_2^* < tr_2+res/2$

where res is the resolution of the translation variables in the parameter space. In this case we say that $(D_1(i,j), D_2(i,j))$ 'votes' for $(rot,expan,tr_1,tr_2)$, i.e., it contributes its weight to $H(rot,expan,tr_1,tr_2)$.

Finally, among the points whose Hough transform exceeds a given threshold in the parameter space, local maxima are found. These represent the hypothetical motions of objects in the image.

3.2 Reliability and Efficiency Considerations

The resolution of the parameter space should be determined by a few considerations: the signal to noise ratio (SNR), the required accuracy, the computation time and the required storage space.

For each independently moving object in the scene, the SNR in the parameter space can be defined as:

(3.4) $$SNR = \frac{\text{no. of votes for the object motion}}{\text{average no. of votes for each parameter value}}$$

(for a different definition see [BRO82]). If the SNR is low, then false peaks, higher than the value corresponding to the object, can be created. Thus the detection of the object's motion, by a straightforward Hough technique, may be difficult or impossible.

Let us assume that the multiplicative parameters of rotation and expansion are quantized into p_1 elements each and that the translation parameters are quantized into

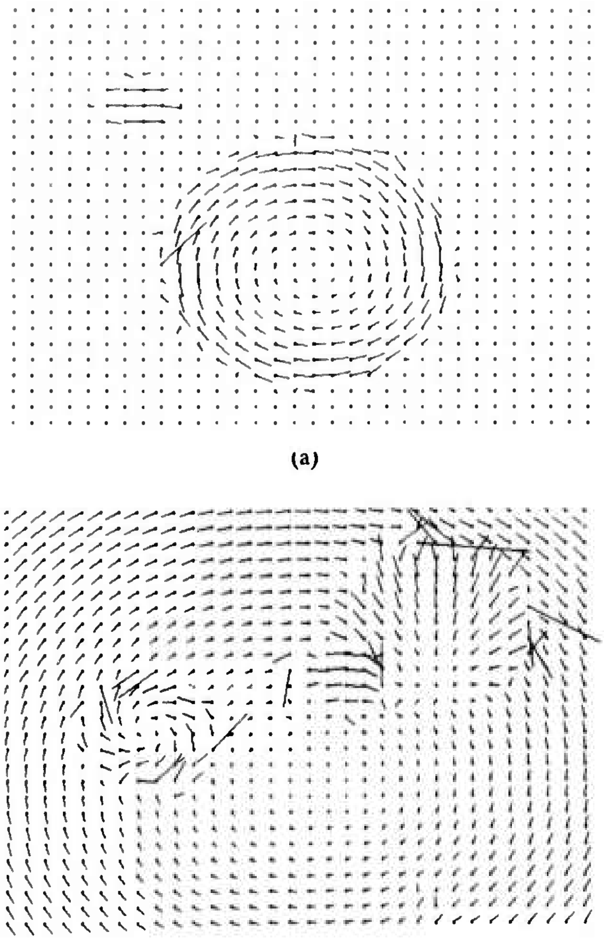


Figure 2: Samples of the displacement fields. (a) First experiment. (b) Second experiment.

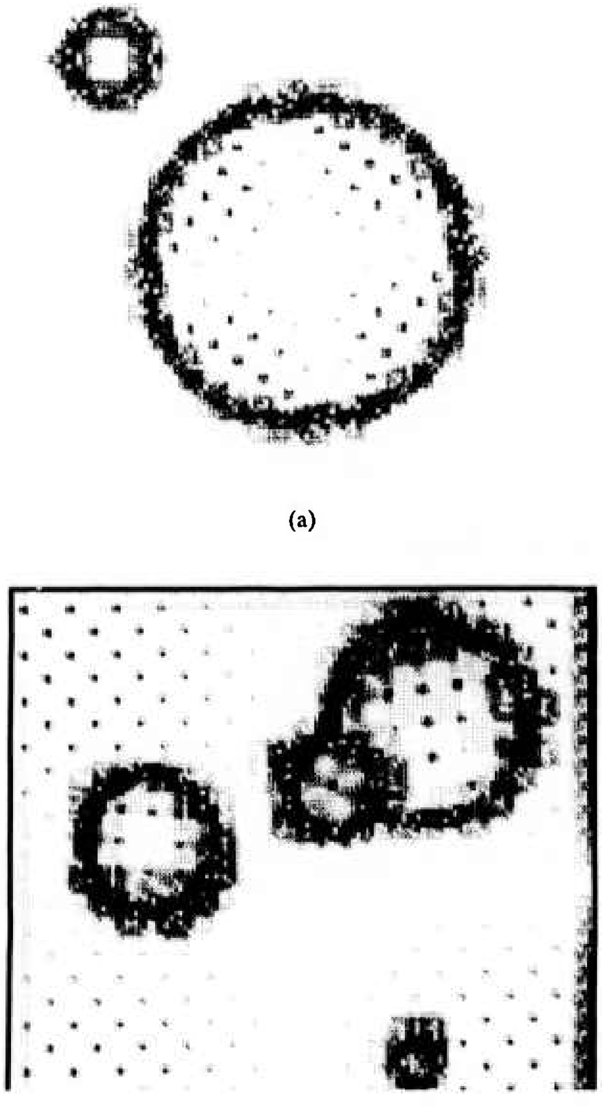


Figure 3: Weight planes. Note the correspondence between low values (represented by darker gray levels) in the weight planes and incorrect values of displacement vectors in the boundaries of the objects. (a) First experiment. (b) Second experiment.

p_2 elements each. Then the parameter space includes $p_1^2p_2^2$ elements. Let n be the number of displacement vectors which are considered in the computation of the Hough transform. According to the voting process described in section (3.1), for each displacement vector and each pair (rot,expan), there exits at most one pair (tr_1,tr_2) of translations such that the displacement vector votes for (rot,expan,tr_1,tr_2). Assuming that tr_1 and tr_2 are likely to be within the limits of the respective dimensions (and that is the case in our experiments) we can estimate the average number of votes for each parameter value as $np_1^2/(p_1^2p_2^2) = n/p_2^2$. If c represents the fraction of the image covered by an object, then it contains cn displacement vectors, where $0<c\leq 1$. Thus, we can estimate the SNR by:

$$\text{(3.5)} \qquad \text{SNR} = \frac{cn}{n/p_2^2} = cp_2^2$$

If for reliable detection of the object, the SNR should be larger than some threshold t, then p_2 should satisfy the constraint $p_2 \geq \sqrt{t/c}$. For example, if t=10 and c=0.01 then p_2 should be at least 32. This observation also indicates that for a given p_2, the motion transformation of a small object may be difficult to detect.

The second consideration is the required accuracy. The parameter resolution can be dynamically modified to fit the expected constraints of the task domain. If, for example, the maximal value of rotation is 1/8 radian, the minimal value is −1/8 radian and the required resolution is 1/128 radian, then p_1 should be at least 32.

The third consideration is computation time. Computationally, the most expensive process is the voting process. We saw in section (3.1) that the basic operation in this process is computing tr_1, tr_2 for each displacement vector and each pair (rot,expan). Therefore, the voting process takes approximately snp_1^2 time units, where each basic operation takes s time units.

The fourth consideration is the required memory for the parameter space which includes $p_1^2p_2^2$ elements. If we combine the requirements for high SNR and high accuracy we may have

$$\text{(3.6)} \qquad p_1^2p_2^2 \geq 32^4 > 1000000$$

In such a large array, finding local maxima also becomes a computationally expensive task.

Finally, assuming that we want to obtain a given accuracy and we are given a storage space with a fixed size, the optimal values of p_1 and p_2 can be determined. Let us suppose that the image contains m^2 pixels and that the origin of the coordinate system is in the center of the image. Then, using a resolution of ϵ_1 in the multiplicative parameters of rotation and expansion can cause, at the boundary of the image, a displacement error of $m\epsilon_1/2$. Therefore, it is reasonable to quantize the parameter space in such a way that $m\epsilon_1 = \epsilon_2$, where ϵ_2 is the resolution of translation. Consequently, if m is multiplied by 4, for example, then p_1 should be multiplied by 2 and p_2 should be divided by 2.

4. Computing Motion Parameters from Displacement Field Information

4.1 Key Ideas

The proposed method is intended to reduce the problems mentioned in the last section and to test mechanisms for solving such problems for even more difficult tasks, e.g. recovering the motion parameters of 3-D motion with six degrees of freedom.

The key ideas which are used for accomplishing this goal are the following:

1) Given a large displacement field (such as the 128×128 array in the experiments), we can compute the motion parameters of large objects by using a coarse resolution field. Such a field can be obtained by uniformly sampling the initial field. In this way, we can considerably reduce the computation time.

2) We can find the motion parameters of a given small object by locating a window around the object and applying the Hough transform only to the displacement vectors contained in this window. Such a window can be located by using a multipass approach (see next section). By focusing our attention to the window, we increase the proportion of the vectors contained in the object, i.e., we increase c in equation (3.5). We can now decrease p_2 and still find the motion parameters of the object. This technique enables us to detect small objects and save time and storage space.

3) Even with a limited memory size, we can find accurate parameter values by iteratively using the Hough technique. In each iteration we quantize the parameter space around the values estimated in the previous iteration, using a finer resolution. Other methods for reducing the required space in Hough techniques can be found in [ORO81, SLO81].

4.2 Description

4.2.1 General

The algorithm is based on repeatedly executing a basic cycle of operations. The input to each cycle includes:

1) A list L of motion transformations which were computed in previous cycles (initially L is an empty list).

2) A binary mask array A in registration with the displacement field. Each element in A is either 1 or 0: 1 if the corresponding displacement vector is consistent with one of the already computed motion transformations; 0 otherwise. Initially all the entries in this array are set to 0.

Each cycle is composed of the following steps:

1) Locate windows in the image which contain relatively dense clusters of 0-entries in A. Initially there is one window consisting of the whole image.

2) For each window compute the Hough transform and hypothesize (see section 4.2.3) the motion transformations.

3) Test, sequentially, the hypothesized transformations. The test is done by considering the 0-entries in A that are contained in the window, and summing the weights of the associated displacement vectors which are consistent with the hypothesized transformation. If the sum is higher than a given threshold, the transformation is confirmed. In this case it is added to the list L and the array A is updated correspondingly.

4.2.2 Locating Windows

A window can be described as a set $\{(i,j): i_0 \leq i < i_1, j_0 \leq j < j_1\}$. For implementation reasons, we consider only windows such that

$$i_0, i_1, j_0, j_1 \in \{0,4,8,...,128\}$$

and

$$i_1 - i_0, j_1 - j_0 \in \{8,16,32,64\}$$

For each window, we define a criterion function CR by:

$$(4.1) \quad CR = \frac{\text{no. of 0-entries of A in the window}}{\sqrt{\text{area of the window}}}$$

We look for windows with high values of CR. Such windows contain dense clusters of 0-entries in A. The use of square root in the denominator of equation (4.1) means that this density can be lower as the window becomes larger. If we would eliminate the square root in this equation, then too small windows, which contain only 0-entries, would be chosen. If we do not find any appropriate windows, i.e. windows whose criterion function exceeds a given threshold, the process is stopped. The reason is that probably there are no more objects whose motion transformation has not already been found.

Figure 4 shows the windows found in the second cycle when the method is applied to the images in figure 1. Figure 5 shows the A arrays when the processes are stopped. The black areas in figure 5, which represent the 0-entries in the A arrays, correspond to incorrect values of displacement vectors in the boundaries of the objects.

4.2.3 The Hypothesizing Phase

Before we start the voting process of the displacement vectors in a given window, we have to decide which vectors take part in this process and how the parameter space is defined. If the window contains no more than 1024 elements, then all of them take part in the voting process. Otherwise, for efficiency considerations (see section 4.1), we will utilize a uniformly sampled subset of 1024 elements. For example, if the window is 32×64 elements, we define a sub-array of 32×32 elements by choosing all the elements (i,j) such that j is even.

The parameter space is an adjustable 4-D array ('adjustable' means that the number of elements in each dimension is not fixed) which contains 17^4 ($\approx$90000) elements. We assume that the rotation is limited to 0.125 radians in each direction, the expansion (or contraction) is limited to 0.125 and the translation is limited to 8 pixels in each direction. The axes which correspond to rotation and expansion contain p_1 elements each and the axes which correspond to the translations contain p_2 elements each. If the length of the window is at least 64 elements then, according to the argument described at the end of section 3.2 for equalizing the effective parameter resolutions, we choose $p_1 = p_2 = 17$; otherwise p_1 is decreased and p_2 is increased. So, for example, if the window is 16×16, we choose $p_1 = 9$ and $p_2 = 31$.

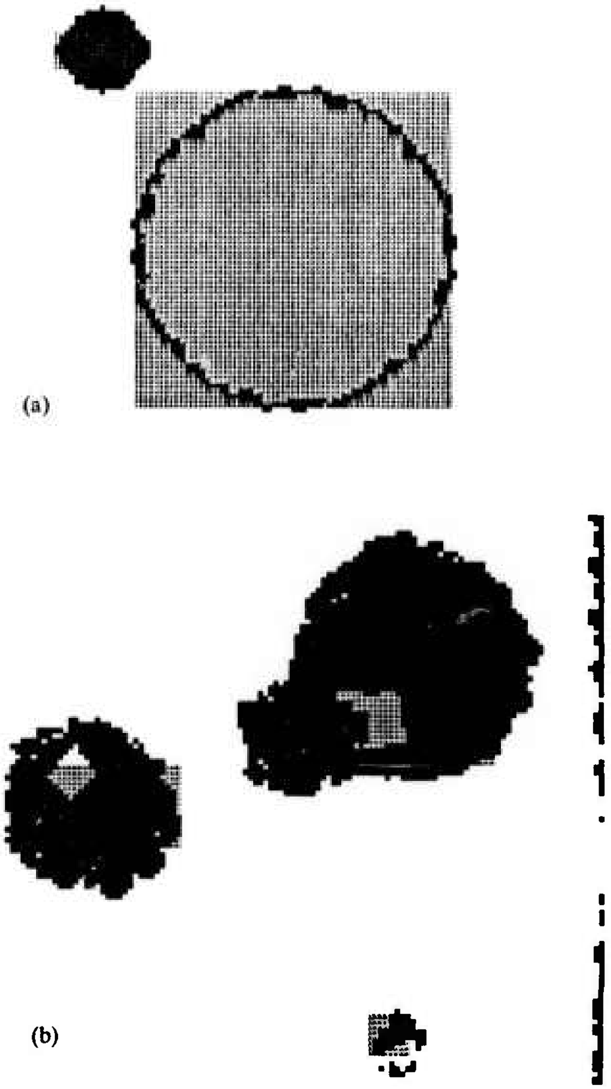


Figure 4: Optimal windows found in the A arrays during the second cycle of the experiments. (a) First experiment. (b) Second experiment.

After the voting process is finished, local maxima in the Hough transform are determined. From these candidates, the ones that exceed a given threshold are selected. The threshold is proportional to the number of all the voting displacement vectors. If the resolution of the translation axes is more than 1/2 pixel, we define a new parameter space, around each maxima point, with finer resolution. We then recompute the Hough transform. The process is repeated until we achieve a resolution of 1/2 pixel at most. At the end of this process we have a set of hypothesized transformations.

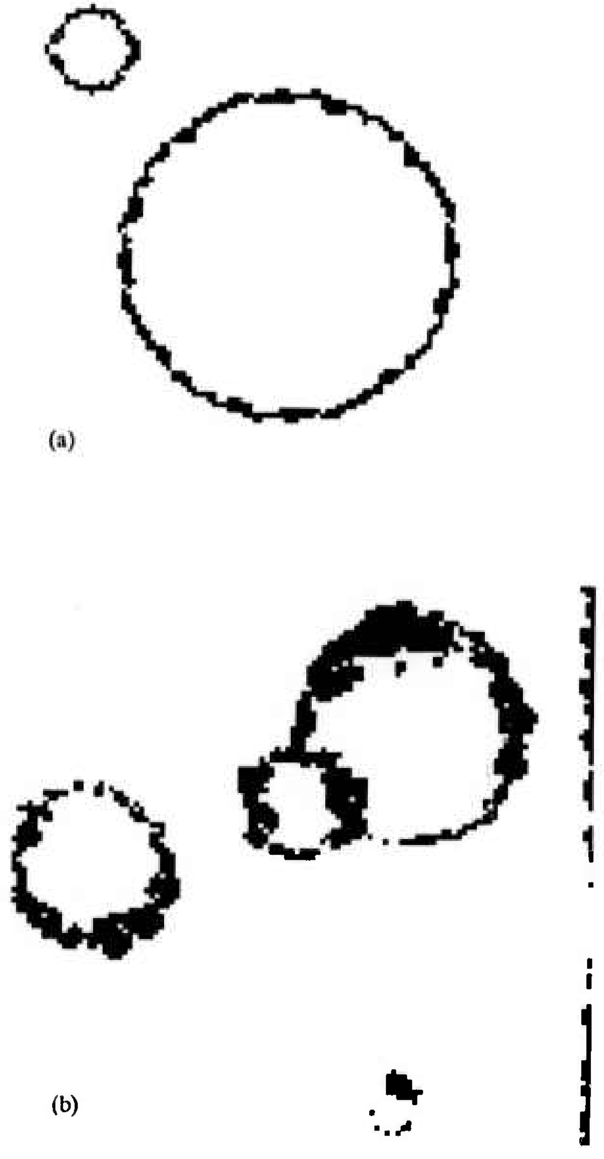


Figure 5: Final A arrays. (a) First experiment. (b) Second Experiment.

4.2.4 The Testing Phase

In this phase we sequentially test the hypothesized transformations in the order of their Hough transform values in the parameter space. We test only the transformations which are still not included in the list L of confirmed transformations. When we test a given transformation, we check all the displacement vectors with associated 0-entry in the corresponding window. We sum the weights of such vectors which are consistent with the hypothesized transformation. We also compute a threshold proportional to the number of 0-entries in A, contained in the window. If the sum is higher than the threshold, we accept the transformation, add it to the list L and update the array A. In the current implementation, the process is stopped if we do not accept any transformation in any of the windows.

5. Experiments

We performed two experiments based on two pairs of 128×128 artificial images (figure 1). In the experiments the objects were transformed according to the upper values in each entry in tables 5.1 and 5.2. The lower numbers in these entries are the computed parameters.

	rotation (radians)	expansion	vertical translation (pixels)	horizontal translation (pixels)
object A				
actual	0.	0.	0.	0.
computed	0.	0.	0.	0.
object B				
actual	0.07	0.	0.	0.
computed	0.0781	0.	0.	0.
object C				
actual	0.	0.	0.	2.
computed	0.	0.	0.	2.

table 5.1 - first experiment

In the first stage – the displacement field was computed (figure 2). Note the errors at the boundaries of the objects which correspond to low values in the weight planes (figure 3).

In experiment 1, during the first cycle of the algorithm for computing the motion parameters, the motion transformations of objects A and B were detected (see the results in table 5.1). In the second cycle, two windows were located around areas in the mask array A with relatively dense clusters of 0-entries (figure 4), but only the window around object C gave a positive result – the motion transformation of object C. In the final cycle no appropriate windows were found in the array A (figure 5).

Corresponding results from experiment 2 are also shown in figure 4 and figure 5. The computed transformations are shown in table 5.2.

	rotation (radians)	expansion	vertical translation (pixels)	horizontal translation (pixels)
object A				
actual	0.025	0.	0.	0.
computed	0.0234	0.	0.	0.
object B				
actual	0.	0.1	−1.5	0.
computed	0.	0.09375	−1.2	0.
object C				
actual	−0.1	0.	0	2.2
computed	−0.09375	0.	−0.2	2.1
object D				
actual	0.12	−0.1	0.	0.
computed	0.125	−0.0937	−0.2	0.
object E				
actual	0.	0.125	−1.1	0.7
computed	0.	0.0625 (*)	−1.05	0.6

table 5.2 - second experiment

(*) The large error indicated in this entry is due to the small size of object E (radius = 8 pixels) which reduces the possible resolution in the measurements of rotation and expansion.

6. Conclusions and Extensions

This work demonstrates an efficient and robust algorithm, based on the Hough technique, for recovering motion parameters in scenes containing several independently moving objects. An hierarchical approach, combined with a windowing scheme, is implemented in order to deal with objects of different size. The storage space and computation time can be limited, while still computing the motion parameters very accurately and distinguishing between real objects and noise effects.

We hope to extend this work for sequences of images and for recovering the 3-D motion parameters of rigid objects. However, the latter task is much more difficult than recovering 2-D motion parameters. In the 2-D case each vector contributes two constraints (equations 2.4 and 2.5) whereas in the 3-D case, assuming that depth information is unknown, each vector contributes only one constraint. Therefore, the signal to noise ratio in the parameter space (section 3.2) is much lower. In addition, we expect to have problems of ambiguity in the interpretation of noisy displacement fields, where a group of motion transformations can be equally consistent with the data. In such cases a probabilistic approach might be more suitable. We also plan to implement less restricted methods for computing a displacement field or other equivalent information, to use the motion information for object-surround separation, and to test the method in real scenes.

Acknowledgements

I would like to acknowledge the support of Ed Riseman, Al Hanson and the VISIONS group at UMASS. I am especially grateful to Daryl Lawton and P. Anandan for their generous help and many fruitful discussions. Ed, Al and P. Anandan made useful suggestions for improving this paper. This research was supported by DARPA Grant N00014-82-K-0464 and ONR Grant N00014-75-C-0459.

References

[BAL81a] Ballard, D.H., "Parameter Networks: Towards a Theory of Low-Level Vision", Proc. of the 7th IJCAI, Vancouver, Canada, 1981.

[BAL81b] Ballard, D.H. and Kimball, O.A., "Rigid Body Motion from Depth and Optical Flow", TR70, Computer Science Dept., U. Rochester, 1981.

[BRO82] Brown, C.M., "Bias and Noise in the Hough Transform I: Theory", TR105, Computer Science Dept., U. Rochester, 1982.

[FEN79] Fennema, C.L. and Thompson, W.B., "Velocity Determination in Scenes Containing Several Moving Objects", Computer Graphics and Image Processing, Vol. 9, 1979.

[HOR80] Horn, B.K.P. and Schunck, B.G., "Determining Optical Flow", MIT A.I. Memo 572, 1980.

[JAY82] Jayaramamurthy, S.N. and Jain, R., "Segmentation of Textured Scenes Using Motion Information", Proc. of the Workshop on Computer Vision, Rindge, N.H., 1982.

[ORO81] O'Rourke, J., "Motion Detection Using Hough Techniques", Proc. of Pattern Recognition and Image Processing, Dalas, Texas, 1981.

[SLO81] Sloan, K.R., "Dynamicallly Quantized Pyramids", Proc. of the 7th IJCAI, Vancouver, Canada, 1981.

[THO81] Thompson, W.B. and Barnard, S.T., "Lower-Level Estimation and Interpretation of Visual Motion", IEEE Computer, August 1981.

STEREO MATCHING FROM THE TOPOLOGICAL VIEWPOINT*

A. Peter Blicher
Artificial Intelligence Laboratory, Computer Science Department, Stanford University
Stanford, California 94305 and
Mathematics Department, University of California, Berkeley

ABSTRACT

A unifying abstract mathematical structure is presented for a number of vision problems, notably stereo, motion stereo, optic flow, and matching. Ideas from modern differential topology are presented and applied to the general matching problem, a common approach to stereo matching, defined as follows. Given 2 picture functions $F_1, F_2 : M^2 \to \mathbf{R}^n$, one finds regions $K_1, K_2 \subset M^2$ and a 1-1 *matching* function $g_\pi : K_1 \to K_2$ such that $F_1 = F_2 \circ g_\pi$. It is shown that *generically* for monochrome pictures ($n = 1$) there is a large infinity of solutions, but for 2 or more color dimensions ($n \geq 2$) the solution is unique.

The paper is offered partially in the hope of introducing vision workers to this type of mathematics and persuading them of its utility.

I INTRODUCTION

The author

Sub 3.

Analogously to the Erlanger Programm, the task of computer vision can be viewed as finding invariants of irradiance functions under the rigid motion group of $\mathbf{R}^3$. This paper describes this structure in the language of modern abstract mathematics, providing a framework for understanding and the possibility of applying powerful methods to resolve fundamental questions. We prove a theorem which says that occlusion-free stereo matching requires at least 2 color dimensions or some knowledge of the imaging geometry.

For reasons of space, the treatment here is abridged and terse. The interested reader can find a more complete exposition, including mathematical details, definitions, and wider discussion in [Blicher 1983].

II THE MATHEMATICAL STRUCTURE

The structure is depicted by the commutative diagram Fig. (*′). The object surface Σ is embedded in $\mathbf{R}^3$ via i. M^3 is a fixed 3 dimensional subset of $\mathbf{R}^3$, and is the domain of definition for the imaging projection π, which maps it to M^2, the 2 dimensional image space. F_1 is the observed image intensity on some closed set K_1 of the image plane. S_1 and K_1 are corresponding visible regions of Σ and the image space M^2, resp.

We assume that the surface Σ admits a function $F : \Sigma \to \mathbf{R}^n$ which describes intrinsic surface features. E.g., for $F : \Sigma \to \mathbf{R}^1$ (i.e. $n = 1$), F represents an *intrinsic* surface brightness or luminance. Thus we ignore the effect of viewpoint on image irradiance, or equivalently, we take the reflectance function to be constant. To the extent that we deal only with small changes in viewpoint, that will usually be a good approximation. F can be thought of as the intrinsic surface property ***albedo***; then our analysis deals with quantities that depend only on albedo, to good approximation. For the case $n \geq 2$, we have in mind color images: normal human cone vision incorporates a function $F_1 : K_1 \to \mathbf{R}^3$ ($n = 3$). Note that we also subsume cases for a smaller (i.e. $n = 2$) or larger ($1 \leq n < \infty$) number of passbands, or in fact any surface attribute, such as a multi-dimensional texture measure, which can be thought of as taking pointwise values in some finite dimensional space.

$$\begin{tikzcd}
\mathbf{R}^3 \arrow[rrrr, "g"] & & & & \mathbf{R}^3 \\
\cup & & & & \cup \\
M^3 \arrow[ddd, "\pi"'] & & \Sigma \arrow[uull, "i"'] \arrow[uurr, "i_g"] & & M^3 \arrow[ddd, "\pi"] \\
 & & \cup \quad \cup & & \\
 & S_1 \arrow[dl] & {} \arrow[dd, "F"] & S_g \arrow[dr] & \\
M^2 & & & & M^2 \\
 & & \mathbf{R}^n & & \\
\cup & & & & \cup \\
K_1 \arrow[uurr, "F_1"'] \arrow[rrrr, "g_\pi"'] & & & & K_g \arrow[uull, "F_g"]
\end{tikzcd}$$

Fig. (*′)

The map $g : \mathbf{R}^3 \to \mathbf{R}^3$ is a rigid motion of $\mathbf{R}^3$. For many surfaces Σ, different viewpoints g have different domains of visibility of Σ. So, in general, g_π as pictured may not be well defined, since we cannot be sure that for K_1, K_g, S_1, S_g as we have defined them, that $S_1 \subset S_g$, or equivalently that $\pi \circ i_g : \Sigma \to M^2$ is 1-1 on S_1. I.e., part of what we see in the picture F_1 might be hidden from view when we look after doing g. Hence the regions K_1, K_g must be chosen in such a way that g_π is well-defined. For example, having chosen S_1, S_g as above, we can define $S'_1 = S'_g = S_1 \cap S_g$ and $K'_1 = \pi \circ i\,(S'_1)$ and $K'_g = \pi \circ i_g(S'_g)$. With these restrictions, g_π is a diffeomorphism $K_1 \to K_g$ with the property that $F_1(p) = F_g(q) = F_g(g_\pi(p))$, which is the same as saying that g_π is a deformation of the picture F_1 into the picture F_g. Note that this observation is also equivalent to asserting that the diagram is commutative (for the F_1, g_π, F_g loop).

We have sidestepped the issues of occlusion, shadowing and photometry. Nevertheless, major parts of the following problems are subsumed in the structure we have presented.

*This work was supported in part by ARPA contracts MDA903-80-C-0102 and N00039-82-C-0250.

- Area correlation stereo
- General matching
- Motion stereo and optical flow
- Feature based stereo
- Singularity tracking

III STEREO BY GENERAL MATCHING

As a straightforward (but not trivial!) application of the abstract viewpoint we are proposing, we show that for monochrome images, the matching problem is insoluble, and study the conditions which allow unique solution.

$$\begin{tikzcd} K_1 \arrow[rr, "g_\pi"] \arrow[dr, "F_1"'] & & K_2 \arrow[dl, "F_2"] \\ & \mathbf{R}^n & \end{tikzcd}$$

Fig. (GM)

A common approach to stereo matching is via general matching. I.e., given 2 picture functions $F_1, F_2 : M^2 \rightarrow \mathbf{R}^n$, one finds regions $K_1, K_2 \subset M^2$ and a 1-1 *matching* function $g_\pi : K_1 \rightarrow K_2$ such that the diagram Fig (GM) commutes. Only after the matching function is found is the surface embedding computed by associating relative depth with relative disparity at each point of (say) K_1. We assume that for the matching phase no information about imaging geometry is used, in particular that there is no assumption of rectification, and no knowledge of epipolar geometry is used, so that arbitrary (but sufficiently differentiable) distortions are possible. Since we are concerned with existence, this is an idealization of what happens in practice, where usually there is at least implicit use of some geometric constraints. We show, in fact, that such use is ***necessary.***

The following question then arises:

Problem (Uniqueness of General Matching). If we seek an arbitrary (piecewise) C^1 diffeomorphism g_π to make Fig.(GM) commute, when are we guaranteed a unique solution to the matching problem?

E.g., if F_1, F_2 are both constant functions, i.e., we have uniformly gray pictures, the problem is completely degenerate, and any diffeomorphism g_π is a solution.

We do not consider problems related to occlusion, and instead assume that in fact it is possible to find regions K_1, K_2 and a map g_π which fulfill the smoothness criteria and make Fig. (GM) commute. Our concern is whether g_π is then unique.

Theorem (2 color theorem). Stereo requires at least 2 color dimensions or 3 space dimensions. I.e., for a monochrome picture, general matching has infinitely many solutions, but for 2 or more color dimensions, it is generally unique. Hence the monochrome case requires knowledge of the imaging situation to constrain the problem. (In particular, this applies to gray-level correlation.)

More precisely, consider the commutative diagram Fig. (GM) where g_π is a C^1 diffeomorphism, F_1, F_2 are C^r, and K_1, K_2 are differentiable submanifolds of $\mathbf{R}^2$.

If $n = 1$ (i.e. the picture is monochrome), then $\exists$ an infinite dimensional family of C^1 diffeomorphisms $\{h_\varphi\}$ such that replacing g_π by h_φ also results in a commutative diagram (i.e. is a solution). The family h_φ is parametrized by (at least) the continuous functions $K_1 \rightarrow \mathbf{R}$, and contains an isomorph of a neighborhood of the identity.

If $n = 2$ (i.e. the picture has 2 color dimensions), then generically there will be a finite number of g_π which make the diagram commute (note we have assumed that such a g_π exists). If we take K_1, K_2 to be rectangles or discs (as in a usual picture) then generically there is a unique g_π.

If $n \geq 3$ (i.e. the picture has at least 3 color dimensions), then generically there will be a unique g_π which makes the diagram commute.

The theorem follows easily from some facts in differential topology. (An excellent introduction to the subject is [Guillemin 1974], and [Hirsch 1976] is a good reference.)

Proof (case $n = 1$: monochrome pictures). The idea of the proof is very simple; the difficulty lies in establishing when it is valid. The idea is this. Observe that if $F_2 \circ g_\pi(p) = F_1(p)$ (i.e. if Fig. (GM) is a commutative diagram) then $g_\pi(F_1^{-1}(x)) = F_2^{-1}(x)$, i.e. g_π takes contour lines to contour lines. Conversely, *any* diffeomorphism $h : K_1 \rightarrow K_2$ which takes contour lines of F_1 to contour lines of F_2 satisfies the conditions for g_π. Thus any g_π taking contour lines to contour lines will solve our local matching problem. But how many such g_π's can there be? Assume for the moment that a typical contour map contains a diffeomorphic image of the fragment represented by the solid lines in Fig. (frag).

If $\psi : K_1 \rightarrow K_1$ is a diffeomorphism leaving contours of F_1 invariant, then if g_π is a matching function so is $g_\pi \circ \psi$. Define ψ as follows. As you go along the dotted line

$$\begin{aligned} \gamma : I &\rightarrow K_1 \\ t &\mapsto \gamma(t) \end{aligned}$$

in Fig (frag), slide each contour along itself by an angle $\theta(t)$. As long as $\theta : I \rightarrow \mathbf{R}$ is a diffeomorphism onto its image, the map ψ will be a diffeomorphism in a neighborhood of the dotted line. To the extent that this picture is valid, there will be as many matchings $g_\pi \circ \psi$ as there are such maps θ.

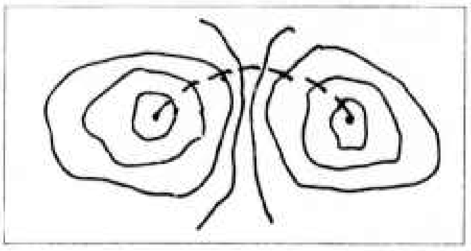


Fig. (frag)

Actually, we are going to use a slightly more general (and technical) method to construct a family of diffeomorphisms ψ_α roughly in 1-1 correspondence with the set of all C^r functions $K_1 \rightarrow \mathbf{R}$. For this we will use a canonical vector field defined along the contour lines of F_1, which will tell us how much to slide each contour line. Define a new vector field on K_1 by rotating each of the local vectors of ∇F_1 by +90°, i.e. +90° counterclockwise, (which is uniquely defined because we have a globally defined inner product on an orientable manifold). One might, e.g. define the new vector field Z on K_1 by $Z(p) = (-b, a)$ if $\nabla F_1(p) = (a, b)$. Note that $Z \cdot \nabla F_1 = 0$ at all p. Since smoothness is defined with respect to coordinates, Z has the same degree of smoothness as ∇F_1. Furthermore, wherever $Z \neq 0$, it is tangent to the contour lines of F_1, so that the orbits of Z are exactly those contour lines, and the critical points are exactly the critical points of ∇F_1. We now consider the flow generated by the vector field Z.

Near the boundary of K_1, the time-one map of this flow may not be defined if a contour line has a boundary. However, this is easily overcome by using a "bump" function [Abraham 1978] $\beta : K_1 \rightarrow \mathbf{R}$ to get a vector field $\beta \cdot Z$ on K_1 which smooth!y goes to zero very close to the boundary, and hence has a flow φ_t which never leaves K_1.

Thus for each t, $\varphi_t : K_1 \rightarrow K_1$ is a diffeomorphism on K_1 leaving contour lines invariant. This family of diffeomorphisms can be enlarged even more. Notice that multiplying the vector field Z by a scalar C^r function $\rho : K_1 \rightarrow \mathbf{R}$ does *not* alter orbits. Therefore we can enlarge the class of diffeomorphisms φ_t by taking all diffeomorphisms $\varphi_{t,\rho}$ given by the flows of $\rho \cdot \beta \cdot Z$ on K_1. Observe that for any constant α, $\varphi_{\alpha t,\rho} = \varphi_{t,\alpha\rho t}$, so if ρ is a constant function, $\varphi_{t,\rho} = \varphi_{\rho t,1} = \varphi_{1,\rho t}$. Thus $\{\varphi_{t,\rho}\} = \{\varphi_{1,\rho}\}$, so by abuse of notation we will write φ_ρ for $\varphi_{1,\rho}$. **QED** ($n = 1$).

A. Discussion of what we have shown so far

In the monochrome matching of 2 regions free of occlusions, the match is far from unique. In fact there are essentially as many matches as C^r functions from such a region to the reals. This stems directly from the fact that the iso-brightness loci constitute connected differentiable 1-dimensional objects. That in turn is a consequence of the fact that the picture is a map from a 2-dimensional object to a 1-dimensional object.

Away from critical points, the matching diffeomorphisms can differ greatly: contour lines can be slid along themselves arbitrarily large amounts. From a practical point of view, given 2 pictures and a matching function, it is a simple matter to choose a ρ and compute φ_ρ, the time-one map of $\rho \cdot \beta \cdot Z$, giving a new match. A matching strategy based on this analysis would first match critical points (generically a discrete combinatorial problem), and then contour lines intersecting the gradients through the critical points. Of course, an actual program would also have to deal with noise, digitization, occlusion, and variation of image irradiance with viewing position; and it would have additional constraints available.

B. The validity of the intuitive picture

Although the theorem is proved for $n = 1$, it's not clear how the intuitive idea of the proof is related to the technical method we actually used. To illuminate that and to disseminate some interesting facts about such mappings, we now turn to the validity of the picture (Fig. (frag)) we presented earlier for the structure of the contour lines. This will require some basic results from differential topology. This is interesting beyond the confines of our present problem; e.g. it casts light on the structure of zero crossings.

First, the proof given above fits into the intuitive scheme presented earlier for using Fig. (frag), since $\rho\beta|Z|$ is essentially the rotation function θ we discussed earlier. But is Fig. (frag) a reasonable picture for the contour lines of a picture function? The following propositions are consequences of the implicit function theorem, Milnor's theorem on 1-manifolds, Sard's theorem, and the genericity of Morse functions (see [Blicher 1983] for details).

1) Almost every level set of a picture is a circle or a line

2) These 1-manifolds account for almost all of the brightness values; the rest are extrema or saddles (critical points).

3) Typically, pictures have isolated critical points (i.e. the critical points do not form blobs, lines, or accumulations).

Now, here's what all this means in terms of Fig. (frag). Choose a picture at random. (Say the picture is bounded by a rectangle R with interior V.) If it has no critical points, then all the level sets are diffeomorphic to (disjoint unions of) line segments (and not circles, which are the only other possibility by Milnor's result, cited above).

Suppose the picture does have critical points. Then "generically" the critical points are isolated.

First let's see what happens near such a critical point. By Morse's Lemma [Guillemin 1974, Hirsch 1976] we know that there is a coordinate system (u, v) in a neighborhood of the critical point p such that $f = f(p) \pm u^2 \pm v^2$. The possible signs correspond to a maximum ($--$), a minimum ($++$), or a saddle ($+-$ or $-+$). So for an extremum, it's easy to see that the level sets are just a point surrounded by circles. For a saddle, the level sets are the sets $u^2 - v^2 = const$, shown in Fig. (saddle). Note that the critical point is isolated (from other critical points), though it is not isolated as part of a level set.

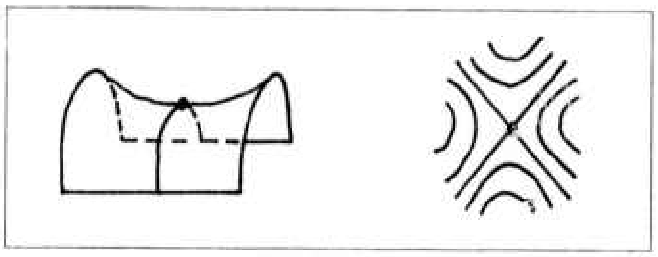


Fig. (saddle)

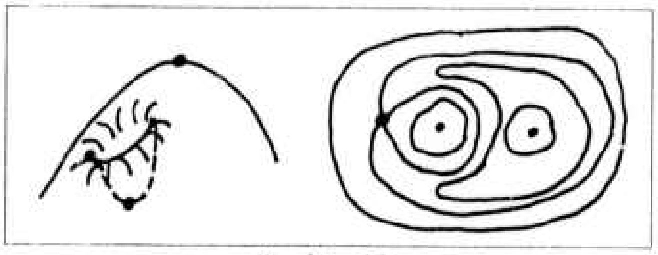


Fig. (dimple)

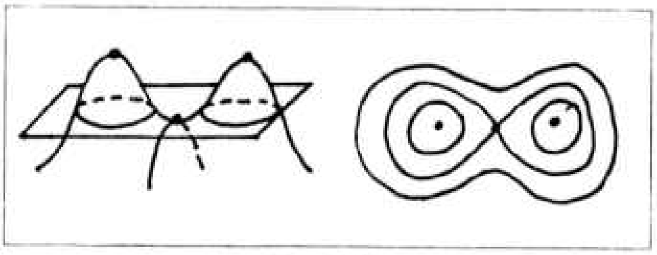


Fig. (pass)

The Morse inequalities tell us that the Euler characteristic is related to the number and type of critical points. In our case, if we assume that the whole region of interest lies within a single circular level set, this means that the number of extrema must be 1 more than the number of saddles. In Fig. (frag), for the rotation directions of the level sets to be consistent with the way we proved the first part of the theorem, we must assume that one of the critical points is a maximum and the other a minimum. But from the Morse inequalities, there must be a saddle somewhere, too. In fact, the larger picture looks like Fig. (dimple), and when there are two maxima (or minima, in Australia), like Fig. (pass).

C. Open dense, usually, generically, almost all, typically

A crucial result used above is that the Morse functions are open dense. This allows us to restrict our attention only to pictures whose critical points are isolated and thus to avoid considering pathological behavior. A property shared by all members of an open dense subset (or a countable intersection of such subsets) is called *generic*, which can be thought of as "most" (see [Blicher 1983, Hirsch 1976, Nitecki 1971, Golubitsky 1973, Guillemin 1974]). Then the (countable) conjunction of generic properties is generic. "Generic" is a key idea in modern differential topology.

D. The cases $n \geq 2$

Let $f : M^m \to M^n$, be C^r and regular at p. The analysis is based on the fact that at a regular point, if there is enough room in the range space, f is a diffeomorphism from a neighborhood U of p to $f(U)$. This is yet another version of the implicit function theorem. The idea of enough room can be made precise simply by requiring the Jacobian to be 1-1. This is the case for a regular point if the dimension of the range space is at least that of the domain space, i.e. if $m \leq n$, which is the situation for us if there are at least 2 color dimensions.

As before, the possible maps g_π which solve the matching problem are exactly those which take level sets to level sets. Since the g_π are diffeomorphisms, we can just study the maps of the level sets of, say, F_1, since they are equivalent by a given g_π to the set of all g_π. (To see this, consider Fig. (equiv). Let h be a diffeomorphism which takes level sets to level sets, i.e. which makes the diagram commutative, and define $g'_\pi = g_\pi \circ h$, so that any h gives us a g'_π. Likewise given such a g'_π, define $h = g_\pi^{-1} \circ g'_\pi$.)

$$\begin{tikzcd}
K_1 \arrow[rr, "g'_\pi"] \arrow[dr, "h"'] \arrow[ddr, "F_1"'] & & K_2 \arrow[ddl, "F_2"] \\
& K_1 \arrow[ur, "g_\pi"'] \arrow[d, "F_1"] & \\
& \mathbf{R}^n &
\end{tikzcd}$$

Fig. (equiv)

First, let's look at how many points can be in $F_1^{-1}(p)$. By the implicit function theorem, since the dimension of the range (i.e. the color space) is at least that of the domain, the level set of a regular value is at most a discrete set of points. Since we are restricting ourselves to compact pictures, the level set must be a finite set (to avoid an accumulation point). Hence on a level set, g_π is constrained to be one of a finite number of permutations of the finite level set. Furthermore, since F_1 is a local diffeomorphism at a regular value, the permutation cannot jump around wildly among neighboring points, so that in fact g_π is a permutation of "sheets."

As it turns out, the higher dimensions are easier to deal with in our context, so we will start with them.

E. Regular points when $n \geq 3$

Theorem. Let M, N be embedded submanifolds of $\mathbf{R}^n$. Then generically, $\dim M + \dim N - n = \dim M \cap N$, where a negative dimension means the intersection is empty.

Locally, the regular sets are embedded submanifolds (by the inverse function theorem), so we can use the preceding to study the inverse images of regular values. In particular, F_I will fail to be 1-1 at places where the embedded regular sets intersect. We are interested in the case that $\dim M = \dim N = 2$, so we see that the intersection is generically of dimension 2, 1, 0, and empty for $n = 2, 3, 4, 5$ resp. Thus if $n \geq 3$ there is no diffeomorphism h other than the identity which makes Fig. (equiv) commute, i.e. such that $F_1 = F_I \circ h$. (Proof: h must be the identity wherever F_1 is 1-1. By the above theorem, that is generically everywhere except on a lower dimensional submanifold. Hence h is the identity on a dense subset, and by continuity uniquely extends to the whole space. QED.) So for the regular points, we have disposed of all the cases of 3 or more color dimensions. Now we look at the singular points, and *their* dimension.

The genericity of Morse functions can be generalized as follows.

Theorem(Critical set dimension). For an open dense subset of $C^\infty_S(M^m, N^n)$, the set of critical points of f where the Jacobian is of rank r

1) comprise a submanifold of M^m

2) $= \emptyset$ if $(m-r)(n-r) > m$

3) is of codimension $(m-r)(n-r)$ in M^m if $(m-r)(n-r) \leq m$
(X is of *codimension* k in Y if $\dim X + k = \dim Y$.)

Before we get involved in studying the critical sets for various color dimensions, we state 2 more closely related theorems which allow us to immediately understand the situations for 4 or more color dimensions. An immediate consequence of the critical set dimension theorem is the

Theorem (Whitney Immersion Theorem). If X, Y are smooth manifolds, with $\dim Y \geq 2 \cdot \dim X$, then maps with no singular points are open dense in $C^\infty(X, Y)$.

For a picture, $\dim X = 2$, so the above theorem applies when there are at least 4 color dimensions. In that case, it states that the typical picture won't have any singularities at all. Hence, typically there is only one "sheet" and no folds.

A further result is the

Theorem (Whitney 1-1 Immersion Theorem). If X, Y are smooth manifolds, with $\dim Y \geq 2 \cdot \dim X + 1$, then 1-1 maps with no singular points are residual (i.e. generic) in $C^\infty(X, Y)$.

So with at least 5 color dimensions, we can assume no color is used twice.

Returning to the critical set dimension theorem, in our case $m = 2$, so what the theorem tells us is that the dimension of the critical set is respectively 1, 0, for $n = 2, 3$, and it is empty for $n \geq 4$.

By reasoning as we did for multiple points of the regular set, h has unique continuation to the critical set for $n \geq 3$, yielding the conclusion that h is generically unique when $n \geq 3$ (for $n = 2$ the 1-1 set need not be dense, so the conclusion wouldn't follow).

To summarize, we have thus far shown that h must be the identity for $n \geq 3$, and is at worst one of a discrete set of sheet permutations for $n = 2$. Now we will pursue the case $n = 2$ a bit further.

F. More about $n = 2$

If we allow the support of a picture to be all of $\mathbf{R}^2$ or S^2, that is all we can say. (Consider, e.g., the function $z \mapsto z^k$ (for some $k \geq 2$) on the complex plane for the picture function. Then the sheets can be permuted leaving the picture invariant.) But a real picture must be finite in extent, so if we are considering subsets of the plane, a rectangle (i.e. a disc) is an appropriate domain to consider. If we are thinking about the sphere, then since we are restricting ourselves to occlusion-free regions, using the entire sphere would imply that there were no observable occlusions, which could only happen in the improbable events that only one object was illuminated, or that the observer could only see an object which completely enclosed him. Right now we are only concerned with the genericity of mappings of the plane, since we are in the context of general matching, so we will make no claims regarding the genericity of occlusion or illumination, though such an analysis is possible.

Let us now assume that the picture support we are considering is topologically a disc. In that case, h, being a homeomorphism, must map the boundary of the disc (a circle S) to itself. If f is 1-1 then h must be the identity. If not, then consider what must happen on this circle. h must be continuable along S, so for $p \in S$, $f^{-1}(p)$ must contain a constant number of points. This excludes the possibility of transverse crossings of $f(S)$. But transverse crossings for such a map are generic, so h must therefore generically be the identity. **QED** ▮

REFERENCES

[Abraham 1978] Abraham,R. and J.E. Marsden, **Foundations of Mechanics**, 2nd ed., Benjamin/Cummings, Reading, Mass., 1978. [QA805.A2 1977, ISBN 0-8053-0102-X].

[Blicher 1983] Blicher, A.P., "Computer Vision from the Topological Viewpoint," Stanford Univ., AI Memo and CS report, spring/summer 1983. (number unavailable at press time.)

[Golubitsky 1973] Golubitsky,M. and V. Guillemin, **Stable Mappings and their Singularities**, (Graduate Texts in Mathematics 14), Springer, New York, 1973. [QA613.64.G64, 516.36, ISBN 0-387-90073-X (soft), ISBN 0-387-90072-1 (hard), ISBN 3-540-90073-X (soft)]

[Guillemin 1974] Guillemin,V. and A. Pollack, **Differential Topology**, Prentice-Hall, Englewood Cliffs, N.J., 1974. [QA613.5.G84, 514'.7, ISBN 0-13-212605-2].

[Hirsch 1976] Hirsch, M.W., **Differential Topology**, Graduate Texts in Mathematics, vol. 33, Springer, New York, 1976.

[Nitecki 1971] Nitecki, Z., **Differentiable Dynamics**, MIT Press, Cambridge, 1971. [ISBN 0-262-14009-8 (hard), ISBN 0-262-64011-2 (paper)].

EXTRACTION OF TEXTURED REGIONS IN AERIAL IMAGERY

H.Y. Lee

Intelligent Systems Group
Departments of Electrical Engineering
and Computer Science
University of Southern California
Los Angeles, California 90089-0272

I. Introduction

In lieu of a complete segmentation approach, we attempt a partial segmentation for natural scenes to extract the large textured regions. The usefulness of texture processing as an early stage in the overall system, however, heavily depends on the nature of the task and the image data. Therefore, as mentioned in an earlier report [1], a way of determining the presence or absence of texture should precede any attempt to perform a texture extraction operation.

A simple method to predict texture presence or absence using a pyramid structure is presented. A new texture measure is defined and used in a region-growing extraction scheme.

II. Prediction of Texture Presence and Uniformity Texture Measures

Local uniformity and the change in uniformity at different resolutions are used as the texture cue in our extraction scheme. While constructing a level of the intensity pyramid where level L is obtained by nonoverlapped block averaging of level L-1, the corresponding levels of the uniformity pyramid indicating the local uniformity at that resolution and of the uniformity-change (UC) pyramid indicating the local uniformity change from the lowest level are computed as shown in Fig. 1. The underlying supposition on which these measures are valid is that the averaging process in constructing a pyramid structure changes a large textured region into a uniform luminance region at the level where the averaging window approximately equals the size of the collection of texture primitives. This is true only if the variations in the illumination and the primitive size are small throughout the region. On the assumption that it is true, we can make the following conjectures:

1. If the given image has a large portion of textured regions, the overall uniformity keeps increasing as the size of the averaging window becomes larger until it exceeds the largest primitive size so that there is no more improvement in uniformity.

2. If the image, on the other hand, is devoid of texture or the textured portion is small, the averaging process may not improve the uniformity or may even decrease it.

The average at each level of the uniformity pyramid (taken over the entire image or a portion of it) is used in determining the presence of large textured regions and estimating the proper level of resolution which is compatible with the size of the texture.

Two test images (Fig. 2) are used to verify the conjecture we made above. These two views of the same scene taken at different seasons have a resolution of 512*.72 pixels. The average values at different levels are shown in Table 1a and Table 1b for Fig. 2a and Fig. 2b respectively. For the strong textural structure of the forested regions in the first view the average value does indeed decrease at level 3 and level 4, while it begins to increase at level 5. From this result, we can not only predict texture presence but also assume that the diameters of the texture primitives lie between 8 and 16. Therefore, we can predict that the best level for texture extraction in the pyramid is level 3 which shows the first significant decrease in the average values. For the second view, which has a much weaker textural structure, the average increases until level 4 and then decreases somewhat at level 5 and level 6. This improvement in uniformity, however, can not be considered as the sign of texture presence, since texture primitives with a diameter between 32 and 64 are very unlikely in an 512*512 aerial image. Though more tests with a variety of images should be made before we confirm the validity of the conjectures, these results show the potential of this simple method. The level 3 uniformity and UC images of Fig. 2a are shown in Fig. 3.

III. Extraction of Textured Regions

Previous approaches to segment an image by texture [2,3,4] were directed to completely divide the image into regions of uniform textural properties without specifically separating the textured portions from the untextured ones. Since the untextured portions of an image can be segmented more easily and accurately using single pixel properties, our approach is to extract connected textured regions one by one and leave the untextured portion untouched for other stages of single pixel processing.

Here, the information from a conservatively extracted region guides the region growing in the next lower level. The resulting region boundary is then refined using the information from the lower level. Therefore, three consecutive levels of the pyramids are involved in extracting compact textured regions. At level T+1, where level T is the one compatible with the texture, compact regions with high uniformity and large uniformity-change are selected as starting elements. At level T, one of the starting elements (magnified by the factor of 2 to take account of the level descent) is grown by merging neighboring pixels whose uniformity and UC values lie inside the uniformity and UC ranges of the magnified element. Though it is unreliable to use the level T-1 uniformity and UC values inside the textured region, a textured region often adjoins untextured regions or regions with a texture of different primitive size, which are detectable at level T-1 (e.g. a forest region touching rivers or roads in an aerial image). At level T-1, therefore, boundary refining is carried out by eliminating the untextured or differently textured portions (with low uniformity or small UC values) from the search area which is constructed at level T and magnified by 2 to be compatible at level T-1. (After the region growing stops at level T, the search area is formed by the exterior boundary pixels as well as boundaries of holes and their neighboring pixels within a distance of 1.) After one region is extracted, the process is repeated using another starting element which is separate from the detected regions. The starting elements from Fig. 2a are shown in Fig. 4 and the binary images of the resulting regions after each step derived from the largest starting element are shown in Fig. 5. Fig. 6 shows the boundaries of the extracted regions on the original image (Fig. 2a). The final result on another test image (high altitude image of the San Francisco urban area) is shown in Fig. 7.

IV. Conclusions

The tests on two aerial images show that the proposed technique can extract large textured regions fairly well. Without the sophisticated description of textures from a stochastic or structural model, simple texture measures achieve sufficient results in certain natural image domain.

V. References

[1] H.Y. Lee and K.E. Price, "Using Texture Edge Information in Aerial Image Segmentation," Technical Report USCISG 101, 1982.

[2] S.C. Carlton and O.R. Mitchell, "Image Segmentation Using Texture and Gray Level," Proc. IEEE Conference on Pattern Recognition and Image Processing, June 1977, pp. 387-391.

[3] P.C. Chen and T.Pavlidis, "Segmentation by Texture Using a Co-Occurrence Matrix and a Split-and-Merge Algorithm," Computer Graphics and Image Processing, Vol. 10, 1979, pp. 172-182.

[4] M. Pietikainen and A. Rosenfeld, "Image Segmentation by Texture using Pyramid Node Linking," IEEE Transactions on Systems, Man, and Cybernetics, December 1981, pp. 822-825.

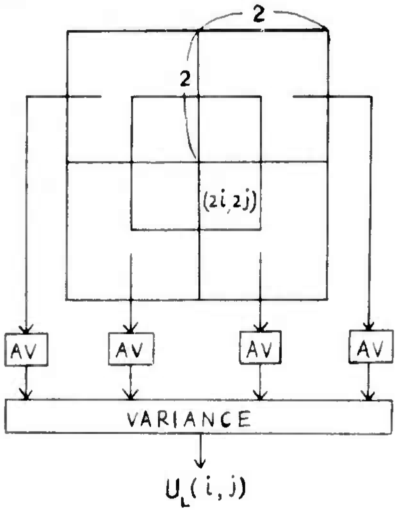


Figure 1. 4 by 4 block at level L-1 involved in the computation of level L features

A(k,ℓ) is the average inside the 2 by 2 block whose lower co-ordinate is (k,ℓ), i.e.,

$$A(k,\ell) = \frac{1}{4} \sum_{m=k-1}^{k} \sum_{n=\ell-1}^{\ell} G_{L-1}(m,n)$$

Level L intensity: $G_L(i,j) = a(2i,2j)$

Level L uniformity:

$$U_L(i,j) = \mathrm{var}\{A(2i-1),2j-1),A(2i-1,2j+1) \\ A(2i+1),2j-1),A(2i+1,2j+1)\}$$

Level L uniformity-change:

$$UC_L(i,j) = \begin{pmatrix}\text{var on the level 0} \\ \text{values inside the whole} \\ \text{block}\end{pmatrix} - U_L(i,j)$$

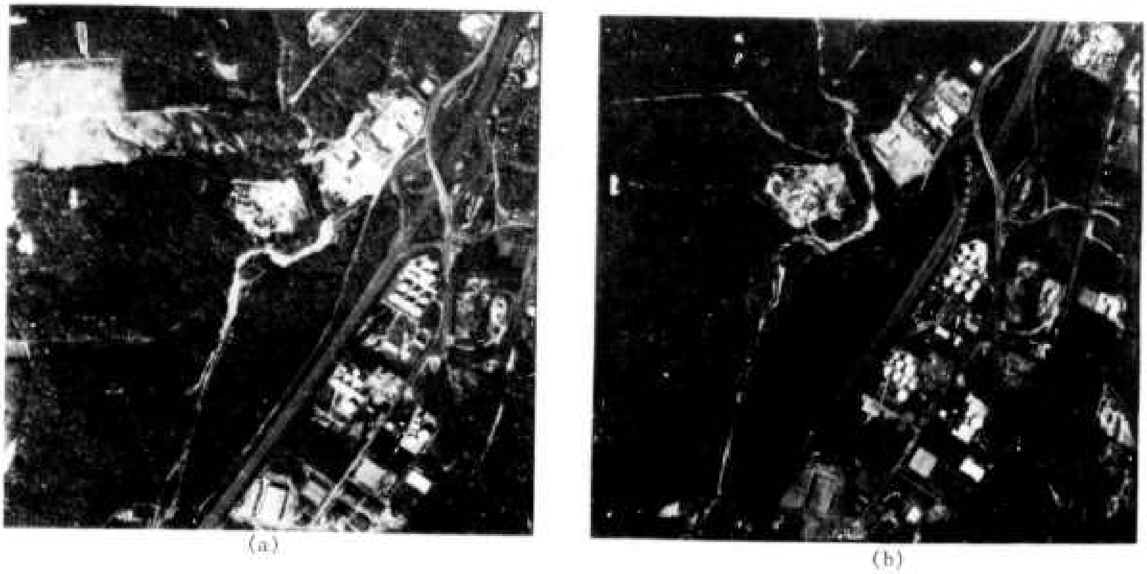


Figure 2. Low altitude aerial image

(a) view 1 (October)
(b) view 2 (August)

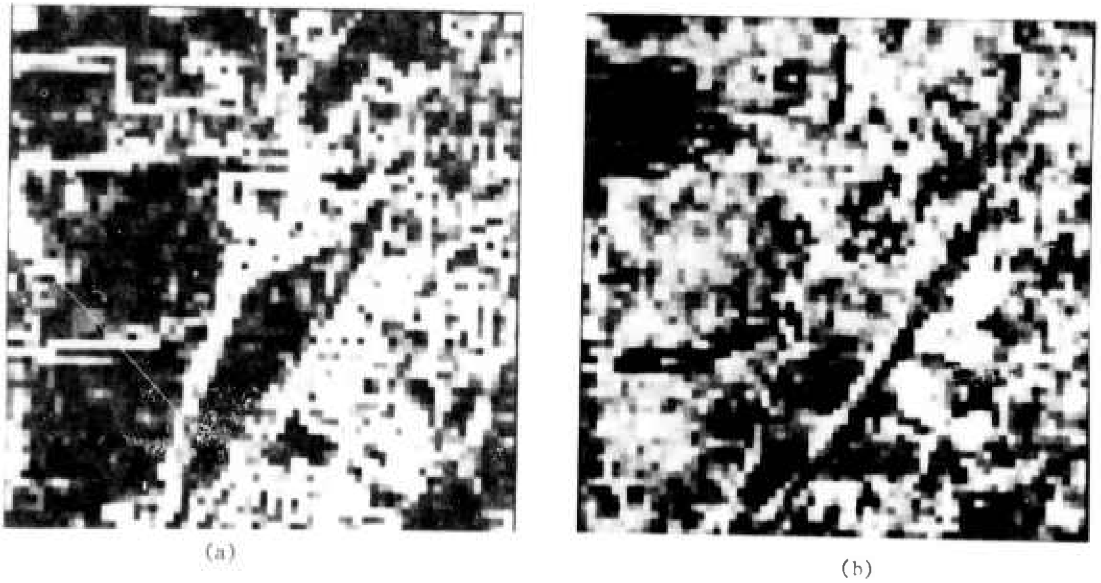


Figure 3. Level 3 images of Fig. 2a

(a) uniformity
(b) uniformity-change

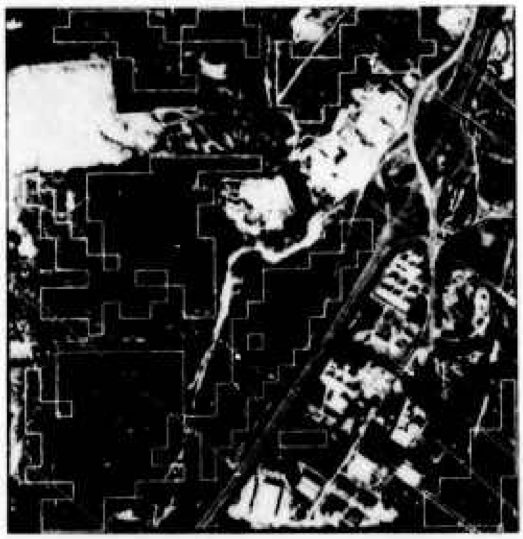
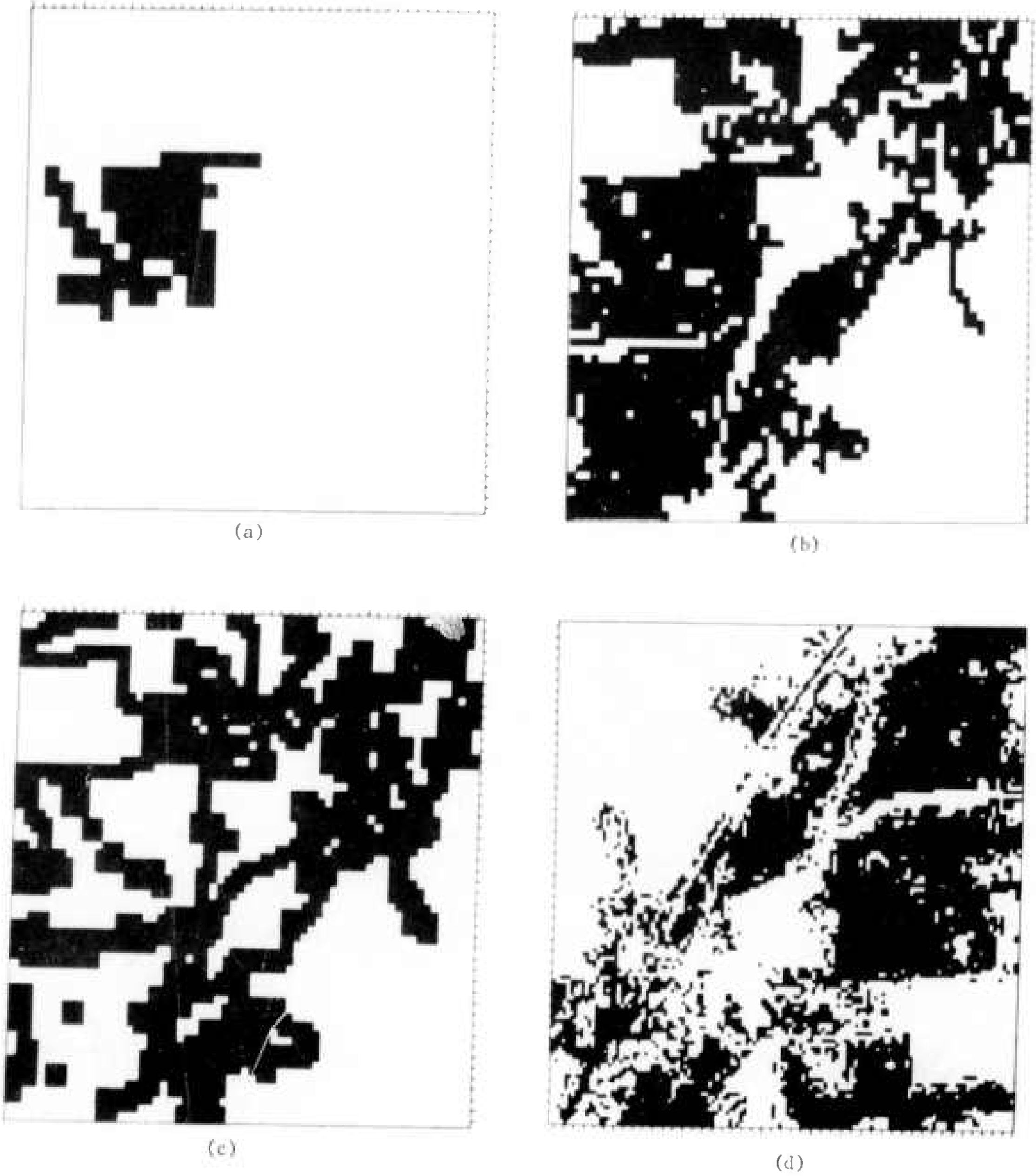


Figure 4. Starting elements selected at level 4



(a)

(b)

(c)

(d)

Figure 5. Results from the largest starting element after each step

(a) starting element at level 4
(b) region grown from (a) at level 3
(c) search area of (b)
(d) after the elimination of the untextured portions at level 2

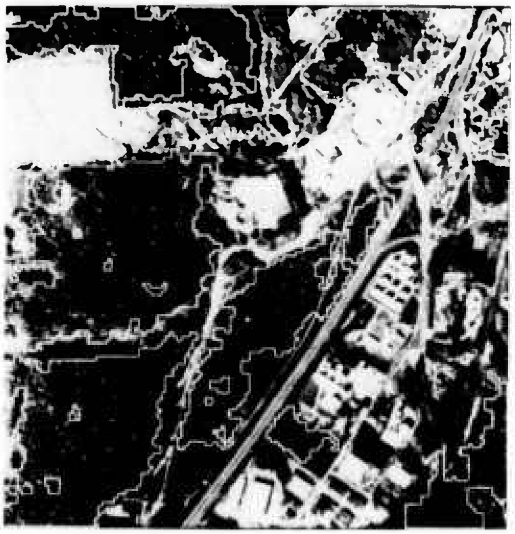


Figure 6. Region boundaries with small noisy regions removed and smoothing

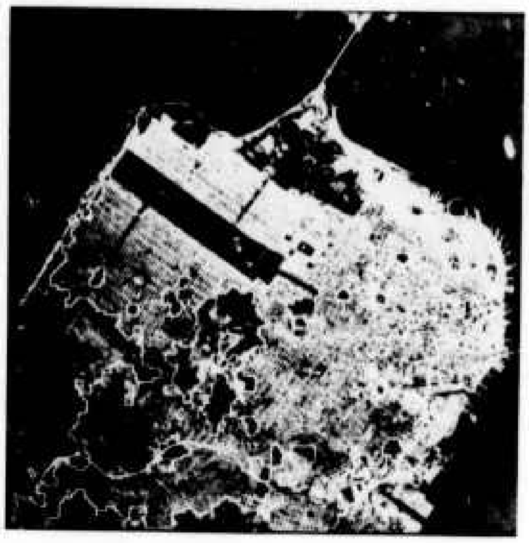


Figure 7. Final result on a high altitude image of San Francisco area

THE TOPOGRAPHIC PRIMAL SKETCH AND ITS APPLICATION TO PASSIVE NAVIGATION

Thomas J. Laffey

Lockheed Palo Alto Research Labs,
0/52-53 B/201, 3251 Hanover Street, Palo Alto, CA 94304

Robert M. Haralick, Layne T. Watson

Departments of Computer Science and Electrical Engineering,
Virginia Polytechnic Institute and State University,
Blacksburg, VA. 24061

ABSTRACT

A complete mathematical treatment is given for describing the topographic primal sketch of the underlying grey tone intensity surface of a digital image. Eacb picture element is independently classified and assigned a unique descriptive label, invariant under monotonically increasing gray tone transformations from the set (peak, pit, ridge, ravine, saddle, flat, and hillside), with hillside having subcategories (inflection point, slope, convex hill, concave bill, and saddle hill). The topographic classification is based on the first and second directional derivatives of tbe estimated image intensity surface. A local, facet model, two-dimensional, cubic polynomial fit is done to estimate the image intensity surface. Zero-crossings of the first directional derivative are identified as locations of interest in the image. Results of the technique applied to digital terrain data and aerial photographs used in the Passive Image Navigation study are presented

1. INTRODUCTION

Representing the fundamental structure of a digital image in a rich and robust way is a primary problem encountered in any general robotics computer-vision system that has to ''understand'' an image. The richness is needed so that shading, highlighting, and shadow information, which are usually present in real manufacturing assembly line situations, are encoded. Richness permits unambiguous object matching to be accomplished. Robustness is needed so that tbe representation is invariant with respect to monotonically increasing gray tone transformations. Current representations involving edges or the primal sketch as described by Marr (1976; 1980) are impoverished in the sense that tbey are insufficient for unambiguous matching. They also do not have the required invariance. Basic research is needed to (1) define an appropriate representation, (2) develop a theory that establishes its relationship to properties that three-dimensional objects manifest on the image, and (3) prove its utility in practice. Until this is done, computer-vision research must inevitably be more ad hoc sophistication than science.

The basis of the topographic primal sketch consists of the classification and grouping of the underlying image intensity surface patches according to the categories defined by monotonic, gray tone, invariant functions of directional derivatives. Examples of such categories are peak, pit, ridge, ravine, saddle, flat, and hillside. From this initial classification, we can group categories to obtain a rich, hierarchical, and structurally complete representation of the fundamental image structure. We call this representation the <u>topographic primal sketch</u>.

Why do we believe that this topographic primal sketch can be the basis for computer vision? We believe it because the light-intensity variations on an image are caused by an object's surface orientation, its reflectance, and characteristics of its lighting source. If any of the three-dimensional intrinsic surface characteristics are to be detected, they will be detected owing to the nature of light-intensity variations. Thus, the first step is to discover a robust representation that can encode the nature of these light-intensity variations, a representation that does not change with strengtb of lighting or with gain settings on the sensing camera. The topograpbic classification does just that. The basic research issue is to define a set of categories sufficiently complete to form groupings and structures that have strong relationships to the reflectances, surface orientations, and surface positions of the three-dimensional objects viewed in the image.

This research has been supported by National Science Foundation grant MCS-8102872. Experiments were carried out witb tbe support of DARPA contract DAAK70-81-C-0098

1.1. The Invariance Requirement

A digital image can be obtained with a variety of sensing-camera gain settings. It can be visually enhanced by an appropriate adjustment of the camera's dynamic range. The gain setting or the enhancing point operator changes the image by some monotonically increasing function that is not necessarily linear. For example, nonlinear enhancing point operators of this type include histogram normalization and equal probability quantization.

In visual perception, exactly the same visual interpretation and understanding of a pictured scene occurs whether the camera's gain setting is low or high and whether the image is enhanced or unenhanced. The only difference is that the enhanced image has more contrast, is nicer to look at, and is understood more quickly by the human visual system.

This fact is important because it suggests that many of the current, low-level computer-vision techniques, which are based on edges, cannot ever hope to have the robustness associated with human visual perception. They cannot have the robustness, because they are inherently incapable ot invariance under monotonic transformations. For example, edges based on zero-crossings of second derivatives will change in position as the monotonic gray tone transformation changes because convexity of a gray tone intensity surface is not preserved under such transformations. However, the topographic categories peak, pit, ridge, valley, saddle, flat, and hillside do have the required invariance.

1.2. Background

Marr (1976) argues tbat the first level of visual processing is the computation of a rich description of gray level changes present in an image, and that all subsequent computations are done in terms of this description, which he calls the primal sketch. Gray-level changes are usually associated with edges, and Marr's primal sketch has, for eacb area of gray level change, a description that includes type, position, orientation, and fuzziness of edge. Marr (1980) illustrates that from this information it is sometimes possible to reconstruct the image to a reasonable degree. Unfortunately, as mentioned earlier, edge is not invariant with respect to monotonic image transformations; besides, it is not a rich enough structure. Difficulty, for example, has been experienced in using edges to accomplish unambiguous stereo matching.

The topographic primal sketch we are discussing as a basis for a representation has the required richness and invariance properties and is very much in the spirit of Marr's primal sketch and the thinking behind Ehricb's relational trees (Ehrich and Foith 1978). Instead of concentrating on gray level changes as edges as Marr does, or on one-dimensional extrema as Ehrich and Foith do, we concentrate on all types of two-dimensional gray level variations. We consider each area on an image to be a spatial distribution of gray levels that constitutes a surface or facet of gray tone intensities having a specific surface shape. It is likely that, if we could describe the shape of the gray tone intensity surface for each pixel, then by assembling all the shape fragments we could reconstruct, in a relative way, the entire surface of the image's gray tone intensity values. The shapes that we already know about that have the invariance property are peak, pit, ridge, ravine, saddle, flat, and hillside, with hillside having noninvariant subcategories of slope, inflection, saddle hillside, convex hillside, and concave hillside.

Knowing that a pixel's surface has the shape of a peak does not tell us precisely where in the pixel the peak occurs; nor does it tell us the height of the peak or the magnitude of the slope around the peak. The topographic labeling, however, does satisfy Marr's (1976) primal sketch requirement in that it contcins a symbolic description of the gray tone intensity changes. Futhermore, upon computing and binding to each topographic label numerical descriptors such as gradient magnitude and direction, directions of the extrema of the second directional derivative along with their values, a reasonable absolute description of each surface shape can be obtained.

1.3. Facet Model

The facet model states that all processing of digital image data has its final authoritative interpretation relative to what the processing does to the underlying gray tone intensity surface. The digital image's pixel values are noisy sampled observations of the underlying surface. Thus, in order to do any processing, we at least have to estimate at each pixel position what this underlying surface is. This requires a model that describes what the general form of the surface would be in the neighborhood of any pixel if there were no noise. To estimate the surface from the neighborhood around a pixel then amounts to estimating the free parameters of the general form. It is important to note that if a different general form is assumed, then a different estimate of the surface is produced. Thus the assumption of a particular general form is necessary and has consequences.

The general form we use is a bivariate cubic. We assume that the neighborhood around each pixel is suitably fit by a bivariate cubic (Haralick 1981;1982). Having estimated this surface around each pixel, the first and second directional derivatives are easily computed by analytic means. The topographic classification of the surface facet is based totally on the first and second directional derivatives. We classify each surface point as peak, pit, ridge, ravine, saddle, flat, or billside, with hillside being broken down further into the subcategories inflection point, convex hill, concave hill, saddle hill, and slope. Our set of topographic labels is complete in the sense that every combination of values of the first and second directional derivative is uniquely assigned to one of the classes.

1.4. Previous Work

Detection of topographic structures in a digital image is not a new idea. There has been a wide variety of techniques to detect (a) peaks and pits (spots), (b) ridges and ravines (lines, streaks), (c) hillsides (edges), and other local features. Some of this work includes Fischler (1982). Lee and Fu (1981), Hsu, Mundy, and Beaudet (1978), Toriwaki and Fukurma (1978), Grender (1976), Paton (1975), Johnston and Rosenfeld (1976), Rosenfeld and Kak (1976) and Peuker and Douglas (1975). Detailed discussion of these methods are beyond the scope of this paper. For an excellent discussion of these works the reader is referred to Laffey (1983).

1.5. A Mathematical Approach

From the investigation of previous work, one can see that a wide variety of methods and labels have been proposed to describe the topographic structure in a digital image. Some of the methods require multiple passes through the image, while others may give ambiguous labels to a pixel. Many of the methods are heuristic in nature. The Hsu, Mundy, and Beudet (1978) approach is the most similar to the one discussed here.

Our classification approach is based on the estimation of the first-and second-order directional derivatives. Thus, we regard the digital-picture function as a sampling of the underlying function f, where some kind of random noise is added to the true function values. To estimate the first and second partials, we must assume some kind of parametric form for the underlying function f. The classifier must use the sampled brightness values of the digital-picture function to estimate the parameters and then make decisions regarding the locations of relative extrema of partial derivatives based on the estimated values of the parameters.

In Section 2, we will discuss the mathematical properties of the topographic structures in terms of the directional derivatives in the continuous surface domain. Because a digital image is a sampled surface and each pixel has an area associated with it, characteristic topographic structures may occur anywhere within a pixel's area. Thus, the implementation of the mathematical topographic definitions is not entirely trivial.

In Section 3 we will discuss the implementation of the classification scheme on a digital image. To identify categories that are local one-dimensional extrema, such as peak, pit, ridge, ravine, and saddle, we search inside the pixel's area for a zero-crossing of the first directional derivative. The directions in which we seek the zero-crossing are along the lines of extreme curvature.

In Section 4, we will discuss the local cubic estimation scheme. In Section 5, we will summarize the algorithm for topographic classification using the local facet model. In Section 6, we will show the results of the classifier on digital terrain data and aerial photographs.

2. THE MATHEMATICAL CLASSIFICATION OF TOPOGRAPHIC STRUCTURES

In this section, we formulate our notion of topographic structures on continuous surfaces and show their invariance under monotonically increasing gray tone transformations. In order to understand the mathematical properties used to define our topographic structures, one must understand the idea of the directional derivative discussed in most advanced calculus books. For completeness, we first give the definition of the directional derivative, then the definitions of the topographic labels. Finally, we show the invariance under monotonically increasing gray tone transformations.

2.1. The Directional Derivative

In two dimensions, the rate of change of a function f depends on direction. We denote the directional derivative of f at the point (r,c) in the direction β by $f_\beta(r,c)$. It is defined as

$$f'_\beta(r,c) = \lim_{h\to 0} \frac{f(r+h*\sin\beta, c+h*\cos\beta) - f(r,c)}{h}.$$

The direction angle β is the clockwise angle from the column axis. It follows directly from this definition that

$$f'_\beta(r,c) = \frac{\partial f}{\partial r}(r,c) * \sin\beta + \frac{\partial f}{\partial c}(r,c) * \cos\beta.$$

We denote the second derivative of f at the point (r,c) in the direction β by $f''_\beta(r,c)$ and it follows that

$$f''_\beta = \frac{\partial^2 f}{\partial r^2}*\sin^2\beta + 2*\frac{\partial^2 f}{\partial r\partial c}*\sin\beta*\cos\beta + \frac{\partial^2 f}{\partial c^2}*\cos^2\beta.$$

The gradient of f is a vector whose magnitude,

$$\left(\left(\frac{\partial f}{\partial r}\right)^2 + \left(\frac{\partial f}{\partial c}\right)^2\right)^{\frac{1}{2}}$$

at a given point (r,c) is the maximum rate of change of f at that point, and whose direction,

$$\tan^{-1}\left(\frac{\frac{\partial f}{\partial r}}{\frac{\partial f}{\partial c}}\right)$$

is the direction in which the surface has the greatest rate of change.

2.2. The Mathematical Properties

We will use the following notation to describe the mathematical properties of our various topographic categories for continuous surfaces. Let

∇f = gradient vector of a function f;

$||\nabla f||$ = gradient magnitude;

$\omega^{(1)}$ = unit vector in direction in which second directional derivative has greatest magnitude;

$\omega^{(2)}$ = unit vector orthogonal to $\omega^{(1)}$;

λ_1 = value of second directional derivative in the direction of $\omega^{(1)}$;

λ_2 = value of second directional derivative in the direction of $\omega^{(2)}$;

$\nabla f \cdot \omega^{(1)}$ = value of first directional derivative in the direction of $\omega^{(1)}$; and

$\nabla f \cdot \omega^{(2)}$ = value of first directional derivative in the direction of $\omega^{(2)}$.

Without loss of generality, we assume $|\lambda_1| >= |\lambda_2|$.

Each type of topographic structure in our classification scheme is defined in terms of the above quantities. In order to calculate these values, the-first and second-order partials with respect to r and c need to be approximated. These five partials are as follows:

$$\frac{\partial f}{\partial r}, \frac{\partial f}{\partial c}, \frac{\partial^2 f}{\partial r^2}, \frac{\partial^2 f}{\partial c^2}, \frac{\partial^2 f}{\partial r \partial c}.$$

The gradient vector is simply $\frac{\partial f}{\partial r}, \frac{\partial f}{\partial c}$. The second directional derivatives may be calculated by forming the Hessian where the Hessian is a 2*2 matrix defined as

$$H = \begin{vmatrix} \frac{\partial^2 f}{\partial r^2} & \frac{\partial^2 f}{\partial r \partial c} \\ \\ \frac{\partial^2 f}{\partial c \partial r} & \frac{\partial^2 f}{\partial^2 f \partial c^2} \end{vmatrix}.$$

Hessian matrices are used extensively in nonlinear programming. Only three parameters are required to determine the Hessian matrix H, since the order of differentiation of the cross partials may be interchanged. That is

$$\frac{\partial^2 f}{\partial r \partial c} = \frac{\partial^2 f}{\partial c \partial r}$$

The eigenvalues of the Hessian are the values of the extrema of the second directional derivative, and their associated eigenvectors are the directions in which the second directional derivative is extremized. This can easily be seen by rewriting f_β'' as the quadratic form

$$f_\beta'' = (\ \sin\beta \ \cos\beta\ ) * H * \begin{vmatrix} \sin\beta \\ \cos\beta \end{vmatrix}.$$

Thus,

$$H\omega^{(1)} = \lambda_1 \omega^{(1)} \text{ and } H\omega^{(2)} = \lambda_2 \omega^{(2)}.$$

Furthermore, the two directions represented by the eigenvectors are orthogonal to one another. Since H is a 2*2 symmetric matrix, calculation of the eigenvalues and cigenvectors can be done efficiently and accurately using the method of Rutishauser (1971). We may obtain the values of the first directional derivative in the direction of either extrema of the second directional derivative by simply taking the dot product of the gradient with the appropriate cigenvector:

$$\nabla f \cdot \omega^{(1)}$$
$$\nabla f \cdot \omega^{(2)}$$

There is a direct relationship between the eigenvalues λ_1 and λ_2 and curvature in the directions $\omega^{(1)}$ and $\omega^{(2)}$: When the first directional derivative $\nabla f \cdot \omega^{(i)} = 0$, then $\lambda_i/(1+(\nabla f \cdot \nabla f))^{1/2}$ is the curvature in the direction $\omega^{(i)}$, i = 1 or 2. For further discussion on the relationship of surface curvature to directional derivative, see Laffey (1983).

Having the gradient magnitude and direction and the eigenvalues and eigenvectors of the Hessian, we can describe the topographic classification scheme.

2.2.1. Peak

A peak (knob) occurs where there is a local maxima in all directions. In other words, we are on a peak if, no matter what direction we look in, we see no point that is as high as the one we are on. The curvature is downward in all directions. At a peak the gradient is zero, and the second directional derivative is negative in all directions. To test whether the second directional derivative is negative in all directions, we just have to examine the value of the second directional derivative in the directions that make it smallest and largest. A point is therefore classified as a peak if it satisfies the following conditions:

$$||\nabla f|| = 0,\ \lambda_1 < 0,\ \lambda_2 < 0.$$

2.2.2. Pit

A pit (sink, bowl) is identical to a peak except that it is a local minima in all directions rather than a local maxima. At a pit the gradient is zero, and the second directional derivative is positive in all directions. A point is classified as a pit if it satisfies the following conditions:

$$\|\nabla f\| = 0,\ \lambda_1 > 0,\ \lambda_2 > 0.$$

2.2.3. Ridge

A ridge occurs on a ridge-line, a curve consisting of a series of ridge points. As we walk along the ridge-line, the points to the right and left of us are lower than the ones we are on. Furthermore, the ridge-line may be flat, slope upward, slope downward, curve upward, or curve downward. A ridge occurs where there is a local maximum in one direction. Therefore, it must have negative second-directional derivative in the direction across the ridge and also a zero first-directional derivative in that same direction. The direction in which the local maximum occurs may correspond to either of the directions in which the curvature is ''extremized'', since the ridge itself may be curved. For nonflat ridges, this leads to the first two cases below for ridge characterization. If the ridge is flat, then the ridge-line is horizontal and the gradient is zero along it. This corresponds to the third case. The defining characteristic is that the second directional derivative in the direction of the ridge-line is zero, while the second directional derivative across the ridge-line is negative. A point is therefore classified as a ridge if it satisfies any one of the following three sets of conditions:

$$\|\nabla f\| \neq 0,\ \lambda_1 < 0,\ \nabla f \cdot \omega^{(1)} = 0$$
or
$$\|\nabla f\| \neq 0,\ \lambda_2 < 0,\ \nabla f \cdot \omega^{(2)} = 0$$
or
$$\|\nabla f\| = 0,\ \lambda_1 < 0,\ \lambda_2 = 0.$$

A geometric way of thinking about the definition for ridge is to realize that the condition $\nabla f \cdot \omega^{(i)} = 0$ means that the gradient direction (which is defined for nonzero gradients) is orthogonal to the direction $\omega^{(i)}$ of extremized curvature.

2.2.4. Ravine

A ravine (valley) is identical to a ridge except that it is a local minimum rather than maximum in one direction. As we walk along the ravine-line, the points to the right and left of us are higher than the one we are on (see Fig. 2). A point is classified as a ravine if it satisfies any one of the following three sets of conditions:

$$\|\nabla f\| \neq 0,\ \lambda_1 > 0,\ \nabla f \cdot \omega^{(1)} = 0$$
or
$$\|\nabla f\| \neq 0,\ \lambda_2 > 0,\ \nabla f \cdot \omega^{(2)} = 0$$
or
$$\|\nabla f\| = 0,\ \lambda_1 > 0,\ \lambda_2 = 0.$$

2.2.5. Saddle

A saddle occurs where there is a local maximum in one direction and a local minimum in a perpendicular direction A saddle must therefore have positive curvature in one direction and negative curvature in a perpendicular direction. At a saddle, the gradient magnitude must be zero and the extrema of the second directional derivative must have opposite signs. A point is classified as a saddle if it satisifies the following conditions:

$$\|\nabla f\| = 0,\ \lambda_1 * \lambda_2 < 0.$$

2.2.6. Flat

A flat (plain) is a simple, horizontal surface, as illustrated in Fig. 3. It, therefore, must have zero gradient and no curvature. A point is classified as a flat if it satisfies the following conditions:

$$\|\nabla f\| = 0,\ \lambda_1 = 0,\ \lambda_2 = 0.$$

Given that the above conditions are true, a flat may be further classified as a foot or shoulder. A foot occurs at that point where the flat just begins to turn up into a hill. At this point, the third directional derivative in the direction toward the hill will be nonzero, and the surface increases in this direction. The shoulder is an analogous case and occurs where the flat is ending and turning down into a hill. At this point, the maximum magnitude of the third directional derivative is nonzero, and the surface decreases in the direction toward the hill. If the third directional derivative is zero in all directions, then we are on a flat, not near a hill. Thus a flat may be further qualified as being a foot or shoulder, or not qualified at all.

2.2.7. Hillside

A hillside point is anything not covered by the previous categories. It has a nonzero gradient and no strict extrema in the directions of maximum and minimum second directional derivative. If the hill is simply a tilted flat (i.e., has constant gradient), we call it a slope. If its curvature is positive (upward), we call it a convex hill. If its curvature is negative (downward), we call it a concave hill. If the curvature is up in one direction and down in a perpendicular direction, we call it a saddle hill.

A point on a hillside is an inflection point if it has a zero-crossing of the second directional derivative taken in the direction of the gradient. The inflection-point class is the same as the step edge defined by Haralick (1982), who classifies a pixel as a step edge if there is some point in the pixel's area having a zero-crossing of the second directional derivative taken in the direction of the gradient.

To determine whether a point is a hillside, we just take the complement of the disjunction of the conditions given for all the previous classes. Thus if there is no curvature, then the gradient must be non zero. If there is curvature, then the point must not be a relative extremum. Therefore, a point is classified as a hillside if all three sets of the following conditions are true ('->' represents the operation of logical implication):

$$\lambda_1 = \lambda_2 = 0 \rightarrow ||\nabla f|| \neq 0,$$

and

$$\lambda_1 \neq 0 \rightarrow \nabla f \cdot \omega^{(1)} \neq 0,$$

and

$$\lambda_2 \neq 0 \rightarrow \nabla f \cdot \omega^{(2)} \neq 0.$$

Rewritten as a disjunction of clauses rather than a conjunction of clauses, a point is classified as a hillside if any one of the following four sets of conditions are true:

$$\nabla f \cdot \omega^{(1)} \neq 0, \; \nabla f \cdot \omega^{(2)} \neq 0$$

or

$$\nabla f \cdot \omega^{(1)} \neq 0, \; \lambda_2 = 0$$

or

$$\nabla f \cdot \omega^{(2)} \neq 0, \; \lambda_1 = 0$$

or

$$||\nabla f|| \neq 0, \; \lambda_1 = 0, \; \lambda_2 = 0.$$

We can differentiate between different classes of hillsides by the values of the second directional derivative. The distinction can be made as follows:

SLOPE	if $\lambda_1 = \lambda_2 = 0$
CONVEX	if $\lambda_1 >= \lambda_2 >= 0, \; \lambda_1 \neq 0$
CONCAVE	if $\lambda_1 <= \lambda_2 <= 0, \; \lambda_1 \neq 0$
SADDLE HILL	if $\lambda_1 * \lambda_2 < 0$

A slope, convex, concave, or saddle hill is classified as an inflection point if there is a zero-crossing of the second directional derivative in the direction of maximum first directional derivative (i.e., the gradient).

2.2.8. Summary of the Topographic Categories

A summary of the mathematical properties of our topographic structures on continuous surfaces can be found in Table 1. The table exhaustively defines the topographic classes by their gradient magnitude, second directional derivative extrema values, and the first directional derivatives taken in the directions which extremize second directional derivatives. Each entry in the table is either 0, +, -, or *. The 0 means not significantly different from zero; + means significantly different from zero on the positive side; - means significantly different from zero on the negative side, and '*' means it does not matter. The label ''Cannot Occur'' means that it is impossible for the gradient to be nonzero and the first directional derivative to be zero in two orthogonal directions.

From the table, one can see that our classification scheme is complete. All possible combinations of first and second directional derivatives have a corresponding entry in the table. Each topographic category has a set of mathematical properties that uniquely determines it.

(Note: Special attention is required for the degenerate case $\lambda_1 = \lambda_2 \neq 0$, where $\omega^{(1)}$ and $\omega^{(2)}$ can be any two orthogonal directions. In this case, there always exists an extreme direction ω which is orthogonal to ∇f, and thus the first directional derivative $\nabla f \cdot \omega$ is always zero in an extreme direction. To avoid spurious zero directional derivatives, we choose $\omega^{(1)}$ and $\omega^{(2)}$ such that $\nabla f \cdot \omega^{(1)} \neq 0$ and $\nabla f \cdot \omega^{(2)} \neq 0$, unless the gradient is zero.)

Table 1. Mathematical Properties of Topographic Structures

$\|\|\nabla f\|\|$	λ_1	λ_2	$\nabla f \cdot \omega^{(1)}$	$\nabla f \cdot \omega^{(2)}$	Label
0	-	-	0	0	Peak
0	-	0	0	0	Ridge
0	-	+	0	0	Saddle
0	0	0	0	0	Flat
0	+	-	0	0	Saddle
0	+	0	0	0	Ravine
0	+	+	0	0	Pit
+	-	-	-,+	-,+	Hillside
+	-	*	0	*	Ridge
+	*	-	*	0	Ridge
+	-	0	-,+	*	Hillside
+	-	+	-,+	-,+	Hillside
+	0	0	*	*	Hillside
+	+	-	-,+	-,+	Hillside
+	+	0	-,+	*	Hillside
+	+	*	0	*	Ravine
+	*	+	*	0	Ravine
+	+	+	-,+	-,+	Hillside
+	*	*	0	0	Cannot Occur

2.3. The Invariance of the Topographic Categories

For a proof on the invariance of the topographic categories {peak, pit, ridge, ravine, saddle, flat, and hillside}, see Haralick, Watson, and Laffey (1983), or Laffey (1983).

2.4 Ridge and Ravine Continuums

The definitions for ridge and ravine can lead to possibly some unexpected results. For example, all points on a right circular cone, except the vertex, will be labeled ridge. Whether one wishes to call these points ridge points or something else is a matter of taste. These points are classified as ridge points because as one walks up the cone toward the vertex the points to the left and right are lower than the one you are on. The continuum of ridges may or may not be acceptable depending upon your viewpoint. Further work by Haralick (forthcoming) has partially solved this problem.

3.0 THE TOPOGRAPHIC CLASSIFICATION ALGORITHM

The definitions of Section 2 cannot be used directly since there is a problem of where in a pixel's area to apply the classification. If the classification were only applied to the point at the center of each pixel, then a pixel having a peak near one of its corners, for example, would get classified as a concave hill rather than as a peak. The problem is that the topographic classification we are interested in must be a sampling of the actual topographic surface classes. Most likely, the interesting categories of peak, pit, ridge, ravine, and saddle will never occur precisely at a pixel's center, and if they do occur in a pixel's area, then the pixel must carry that label rather than the class label of the pixel's center point. Thus one problem we must solve is to determine the dominant label for a pixel given the topographic class label of every point in the pixel. The next problem we must solve is to determine, in effect, the set of all topographic classes occurring within a pixels's area without having to do the impossible brute-force computation.

For the purpose of solving these problems, we divide the set of topographic labels into two subsets: (1) those that indicate that a strict, local, one-dimensional extremum has occurred (peak, pit, ridge, ravine, and saddle) and (2) those that do not indicate that a strict, local, one-dimensional extremum has occurred (flat and hillside). By <u>one-dimensional</u>, we mean along a line (in a particular direction). A strict, local, one-dimensional extremum can be located by finding those points within a pixel's area where a zero-crossing of the first directional derivative occurs.

So that we do not search the pixel's entire area for the zero-crossing, we only search in the directions of extreme second directional derivative, $\omega^{(1)}$ and $\omega^{(2)}$. Since these directions are well aligned with curvature properties, the chance of overlooking an important topographic structure is minimized, and, more importantly, the computational cost is small.

When $\lambda_1 = \lambda_2 \neq 0$, the directions $\omega^{(1)}$ and $\omega^{(2)}$ are not uniquely defined. We handle this case by searching for a zero-crossing in the direction given by $H^{-1} * \nabla f$. This is the <u>Newton direction</u>, and it points directly toward the extremum of a quadratic surface.

For inflection-point location (first derivative extremum), we search along the gradient direction for a zero-crossing of second directional derivative. For one-dimensional extrema, there are four cases to consider: (1) no zero-crossing, (2) one zero-crossing, (3) two zero-crossings, and (4) more than two zero-crossings. The next four sections discuss these cases.

3.1. <u>Case One</u>: <u>No Zero-Crossing</u>

If no zero-crossing is found along either of the two extreme directions within the pixel's area, then the pixel cannot be a local extremum and therefore must be assigned a label from the set (flat or hillside). If the gradient is zero, we have a flat. If it is nonzero, we have a hillside. If the pixel is a hillside, we classify it further into (inflection point, slope, convex hill, concave hill, or saddle hill). If there is a zero-crossing of the second directional derivative in the direction of the gradient within the pixel's area, the pixel is classified as an inflection point. If no such zero-crossing occurs, the label assigned to the pixel is based on the gradient magnitude and Hessian eigenvalues calculated at the center of the pixel, local coordinates (0,0), as in Table 2.

3.2. <u>Case Two</u>: <u>One Zero-Crossing</u>

If a zero-crossing of the first directional derivative is found within the pixel's area, then the pixel is a strict, local, one-dimensional extremum and must be assigned a label from the set (peak, pit, ridge, ravine, or saddle). At the location of the zero-crossing, the Hessian and gradient are recomputed, and if the gradient magnitude at the zero-crossing is zero, Table 3 is used.

If the gradient magnitude is nonzero, then the choice is either ridge or ravine. If the second directional derivative in the direction of the zero-crossing is negative, we have a ridge. If it is positive, we have a ravine. If it is zero, we compare the function value at the center of the pixel, f(0,0), with the function value at the zero-crossing, f(r,c). If f(r,c) is greater than f(0,0), we call it a ridge, otherwise we call it a ravine.

3.3. <u>Case Three</u>: <u>Two Zero-Crossings</u>

If we have two zero-crossings of the first directional derivative, one in each direction of extreme curvature, then the Hessian and gradient must be recomputed at each zero-crossing. Using the procedure decribed in Section 3.2, we assign a label to each zero-crossing. We call these labels LABEL1 and LABEL2. The final classification given the pixel is based on these two labels and is given in Table 4.

If both labels are identical, the pixel is given that label. In the case of both labels being ridge, the pixel may actually be a peak, but experiments have shown that this case is rare. An analogous argument can be made for both labels being ravine. If one label is ridge and the other ravine, this indicates we are at or very close to a saddle point, and thus the pixel is classified as a saddle. If one label is peak and the other ridge, we choose the category giving us the "most information," which in this case is peak. The peak is a local maximum in all directions, while the ridge is a local maximum in only one direction. Thus, peak conveys more information about the image surface. An analogous argument can be made if the labels are pit and ravine. Similarly, a saddle gives us more information than a ridge or valley. Thus, a pixel is assigned saddle if its zero-crossings have been labeled ridge and saddle or ravine and saddle.

It is apparent from Table 4 that not all possible label combinations are accounted for. Some combinations, such as peak and pit, are omitted because of the assumption that the underlying surface is smooth and sampled frequently enough that a peak and pit will not both occur within the same pixel's area. If such a case occurs, our convention is to choose arbitrarily one of LABEL1 or LABEL2 as the resulting label for the pixel.

Table 2. Pixel Label Calculation for Case One: No Zero-Crossing

$\|\nabla f\|$	λ_1	λ_2	Label
0	0	0	Flat
+	–	–	Concave Hill
+	–	0	Concave Hill
+	–	+	Saddle Hill
+	0	0	Slope
+	+	–	Saddle Hill
+	+	0	Convex Hill
+	+	+	Convex Hill

Table 3. Pixel Lable Calculation for Case Two: One Zero-Crossing

$\|\nabla f\|$	λ_1	λ_2	Label
0	–	–	Peak
0	–	0	Ridge
0	–	+	Saddle
0	+	–	Saddle
0	+	0	Ravine
0	+	+	Pit

Table 4. Final Pixel Classification, Case Three: Two Zero-Crossings

LABEL1	LABEL2	Resulting Label
Peak	Peak	Peak
Peak	Ridge	Peak
Pit	Pit	Pit
Pit	Ravine	Pit
Saddle	Saddle	Saddle
Ridge	Ridge	Ridge
Ridge	Ravine	Saddle
Ridge	Saddle	Saddle
Ravine	Ravine	Ravine
Ravine	Saddle	Saddle

3.4. Case Four: More than Two Zero-Crossings

If more than two zero-crossings occur within a pixel's area, then in at least one of the extrema directions there are two zero-crossings. If this happens, we choose the zero-crossing closest to the pixel's center and ignore the other. If we ignore the further zero-crossings, then this case is identical to case 3. This situation has yet to occur in our experiments.

4.0 SURFACE ESTIMATION

In this section we discuss the estimation of the parameters required by the topographic classification scheme of Section 2 using the local cubic facet model (Haralick 1981). It is important to note that the classification scheme of Section 2 and the algorithm of Section 3 are independent of the method used to estimate the first-and second-order partials of the underlying digital image-intensity surface at each sampled point. Results from using basis functions other than the bi-cubic polynomial are presented in (Laffey 1983). In these experiments the cubic model performed best.

4.1. Local Cubic Facet Model

In order to estimate the required partial derivatives, we perform a least-squares fit with a two-dimensional surface, f, to a neighborhood of each pixel. It is required that the function f be continuous and have continuous first-and second-order partial derivatives with respect to r and c in a neighborhood around each pixel in the rc plane.

We choose f to be a cubic polynomial in r and c expressed as a combination of discrete orthogonal polynomials. The function f is the best discrete least-squares polynomial approximation to the image data in each pixel's neighborhood. More details can be found in Haralick's paper (1981), in which each coefficient of the cubic polynomial is evaluated as a linear combination of the pixels in the fitting neighborhood.

To express the procedure precisely and without reference to a particular set of polynomials tied to neighborhood size, we will canonically write the fitted bicubic surface for each fitting neighborhood as

$$\begin{aligned} f(r,c) = {} & k_1 + k_2 r + k_3 c \\ & + k_4 r^2 + k_5 rc + k_6 c^2 \\ & + k_7 r^3 + k_8 r^2 c + k_9 rc^2 + k_{10} c^3, \end{aligned}$$

where the center of the fitting neighborhood is taken as the origin. It quickly follows that the needed partials evaluated at local coordinates (r,c) are

$$\frac{\partial f}{\partial r} = k_2 + 2k_4 r + k_5 c + 3k_7 r^2 + 2k_8 rc + k_9 c^2$$

$$\frac{\partial f}{\partial c} = k_3 + k_5 r + 2k_6 c + k_8 r^2 + 2k_9 rc + 3k_{10} c^2$$

$$\frac{\partial^2 f}{\partial r^2} = 2k_4 + 6k_7 r + 2k_8 c$$

$$\frac{\partial^2 f}{\partial c^2} = 2k_6 + 2k_9 r + 6k_{10} c$$

$$\frac{\partial^2 f}{\partial r \partial c} = k_5 + 2k_8 r + 2k_9 c$$

It is easy to see that if the above quantities are evaluated at the center of the pixel where local coordinates (r,c) = (0,0), only the constant terms will be of significance. If the partials need to be evaluated at an arbitrary point in a pixel's area, then a linear or quadratic polynomial value must be computed.

5. SUMMARY OF THE TOPOGRAPHIC CLASSIFICATION SCHEME

The scheme is a parallel process for topographic classification of every pixel which can be done in one pass through the image. At each pixel of the image, the following four steps need to be performed

1. Calculate the fitting coefficients, k_1 through k_{10}, of a two-dimensional cubic polynomial in an n-by-n neighborhood around the pixel. These coefficients are easily computed by convolving the appropriate masks over the image.

2. Use the coefficients calculated in step 1 to find the gradient, gradient magnitude, and the eigenvalues and eigenvectors of the Hessian at the center of the pixel's neighborhood, (0,0).

3. Search in the direction of the eigenvectors calculated in step 2 for a zerocrossing of the first directional derivative within the pixel's area. (If the eigenvalues of the Hessian are equal and non-zero, then search in the Newton direction.)

4. Recompute the gradient, gradient magnitude, and values of second directional derivative extrema at each zero-crossing. Then apply the labeling scheme as described in Sections 3.1---3.4.

6. RESULTS

In this section, we show the results of the topographic classification on some digital terrain imagery and aerial photographs used in the Passive Image Navigation Study.

6.1 Results on Digtial Terrain Data

In Figure 1 we show the results of the topographic classification algorithm on digitial terrain data which represents a roughly 4x17 mile strip of land and ocean just east of Monterey, California. The actual image resolution is 121x512 pixels. In Figure 1 we show the results of the labeling for several of the categories. The topmost shows the ravines in white, the next shows the riges, and then the peaks are shown. On the bottom the original grey-level picture is shown.

The algorithm shows excellent results on the digital terrain data. The ridges and ravines appear to be robust enough for use in a reference topographic landmark database. Sensed topography would be matched against the reference database for navigation purposes.

6.2 Results on Aerial Photographs

In figure 2 and 3 we show the results of the classifier on a set of aerial photographs. it seems evident that ravines, ridges, and hillsides (slopes) could serve as reference data in an intensity landmark database. Exactly which topographic categraphic are reliable and how they should be linked together and pruned remains a topic of future research.

7. CONCLUSIONS

In this paper, we have given a precise mathematical description of the various topographic structures that which occur in a digital image and have called the classified image the topographic primal sketch. Our set of topographic categories is invariant under gray tone, monotonically increasing transformations and consists of (peak, pit, ridge, ravine, saddle, flat, and hillside), with hillside being broken down further into the subcategories inflection point, slope, convex hill, concave hill, and saddle hill. The hillside subcategories are not invariant under the monotonic transformations.

The topographic label assigned a pixel is based on the pixel's first-and second-order directional derivatives. We use a two-dimensional cubic polynomial fit based on the local facet model to estimate the directional derivatives of the underlying gray tone intensity surface. The calculation of the extrema of the second directional derivative can be done efficiently and stably by forming the Hessian matrix and calculating its eigenvalues and their associated eigenvectors. Strict, local, one-dimensional extrema (such as pit, peak, ridge, ravine, and saddle) are found by searching for a zero-crossing of the first directional derivative in the directions of extreme second directional derivative (the eigenvectors of the Hessian). We have also identified another direction of interest, the Newton direction, which points toward the extremum of a quadratic surface. The classification scheme was found to give satisfactory results on a number of test images.

7.1. Directions for Further Research

Further research on the topographic primal sketch needs to be done to (1) develop better basis functions, (2) make use of fitting error, (3) find a solution for the ridge (ravine) continuum problem, and (4) develop techniques for grouping of the topographic structures. Basis functions worth considering include trigonometric polynomials, polynomials of higher order, and piecewise polynomials of lower order than cubic. The basis functions problem is to find a set of basis functions and an associated inner product for least-squares approximation that can correctly replicate all common image surface features and be simultaneously computationally efficient and numerically stable. Fitting error needs to be used in deciding into which class a pixel falls. Noise causes the fitting error to increase, and increased fitting error increases the uncertainty of the labeling. Also, global knowledge of how the topographic structures fit together could be used to correct the misclassification error caused by noise. The way the neighborhood size affects the surface fitting error and the classification scheme needs to be investigated in detail.

The ridge (ravine) continuum problem needs to be solved. It may be that there is no way to distinguish between a true ridge and a ridge continuum using only the values of partial derivatives at a point. The solution may require complete use of the partial derivatives in a local area about the pixel.

Most important for the use of the primal sketch in a general robotics computer vision system is the development of techniques for grouping and assembling topographically labeled pixels to form the primitive structures involved in higher-level matching and correspondence processes. How well can stereo correspondence or frame-to-frame time-varying image correspondence tasks be accomplished using the primitive structures in the topographic primal sketch? How effectively can the topographic sketch be used in undoing the confounding effects of shading and shadowing? How well will the primitive structures in the topographic sketch perform in the two-dimensional to three-dimensional object matching process?

REFERENCES

[1] Ehrich, R. W., and Foith, J. P. 1978. Topology and Semantics of Intensity Arrays. Computer vision systems, New York: Academic, pp. 111 -- 128.

[2] Fischler, M. A. 1982. Linear delination, Proceedings Image Understanding Workshop, Palo Alto, California, pp 274-282.

[3] Grender, G. C. 1976. TOPO III: A Fortran Program for Terrain Analysis. Comput. Geosci., London:Pergamon Press 2, pp. 195 -- 209.

[4] Haralick, R. M. 1980. edge and region analysis for digital image data. Comput. Graphics Image Processing, New York:Academic, 12(1):60 -- 73.

[5] Haralick, R. M. 1981. The Digital Edge. Proc. 1981 Conf. Pattern Recognition Image Processing. New York: IEEE Computer Society, pp. 285 -- 294.

[6] Haralick, R. M. 1982. Zero-crossing of second directional derivative edge operator. SPIE Proc. Robot Vision, Bellingham:SPIE

[7] Haralick, R. M. Watson, L.T. and Laffey, T. J. 1983. The topographical primal sketch. Int'l Journal of Robotics Research, MIT press, Cambridge, MA (forthcoming)

[8] Haralick, R. M. 1983. Ridges and valley in a digital image. Comp. Graphics and Image Proc., New York:Academic Press forthcoming

[9] Hsu, S., Mundy, J. L. and Beaudet, P. R. 1978. Web representation of image data. 4th Int. Joint Conf. Pattern Recognition. New York:IEEE Computer Society pp. 675 -- 680.

[10] Johnston, E. G. and Rosenfeld, A. 1975. Digital detection of pits, peaks, ridges, and ravines. IEEE Trans. Syst., Man, Cybern. New York:IEEE Trans.Syst., Man, Cybern. 472 -- 480.

[11] Laffey, T. J., Haralick R. M. and Watson, L. T. 1982. Topographic classification of digital image intensity surfaces. Proc. IEEE Workshop Comput. Vision: Theory Contr. New York: IEEE Computer Society. pp. 171 -- 177.

[12] Laffey, T.J. Topographical classification of digital image intensity surfaces. M.S. Thesis, Dept. of Computer Science, Virginia Polytechnic Institute and State University, February, 1983.

[13] Lee, H. C., and Fu, K. S. 1981. The GLGS image representation and its application to preliminary segmentation and pre-attenetive visual search. Proc. 1981 Conf. Pattern Recognition Image Processing. New York:IEEE Computer Society pp. 256 -- 261.

[14] Marr, D. 1976. Early processing of visual information', Philosophical Trans. Royal Soc. London, London:Royal Society, B 275:483 -- 524.

[15] Marr, D. 1980. Visual information processing: The structure and creation of visual representations. Philosophical Tran. Royal Soc. London, London:Royal Society, B 290:199 -- 218.

[16] Paton, K. 1975. Picture description using Legendre polynomials. Comput. Graphics Image Processing, New York:Academic 4(1):40 -- 54.

[17] Peuker, T. K., and Johnston, E. G. 1972 (Nov.). Detection of surface-specific points by local parallel processing of discrete terrain elevation data. Tech. Rept. 206. College Park: University of Maryland Computer Science Center.

[18] Peuker, T. K. and Douglas, D. H. 1975. Detection of surface-specific points by local parallel processing of discrete terrain elevation data Comput. Graphics Image Processing, New York:Academic 4(4):375 - 387.

[19] Rosenfeld, A., and Kak, A.C. 1976. Digital Picture Processing, Academic Press, New York, New York, PP. 306 - 316.

[20] Rutishauser, H. 1971. Jacobi method for real symmetric matrix. Handbook for automatic computation, volume II, linear algebra, ed. J. H. Wilkinson and C. Reinsch. New York:Springer-Verlag.

[21] Strang, G. 1980. Linear algebra and its applications, 2rd ed. New York:Academic, pp.243 - 249.

[22] Toriwaki, J., and Fukumura, T. 1978. Extraction of structural information from grey pictures. Comput. Graphics Image Processing, New York:Academic, 7(1):30 - 51.

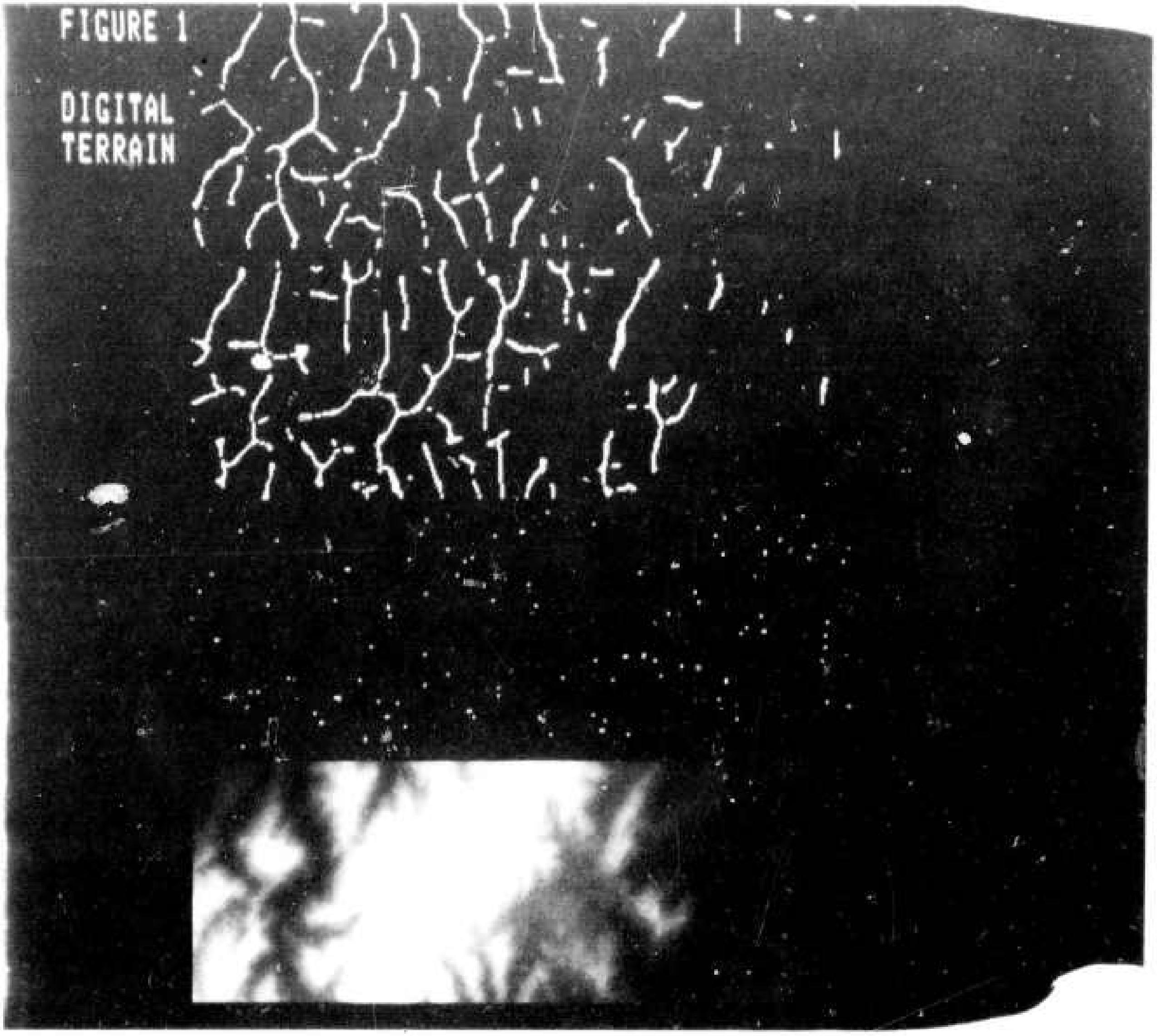


FIGURE 1

DIGITAL TERRAIN

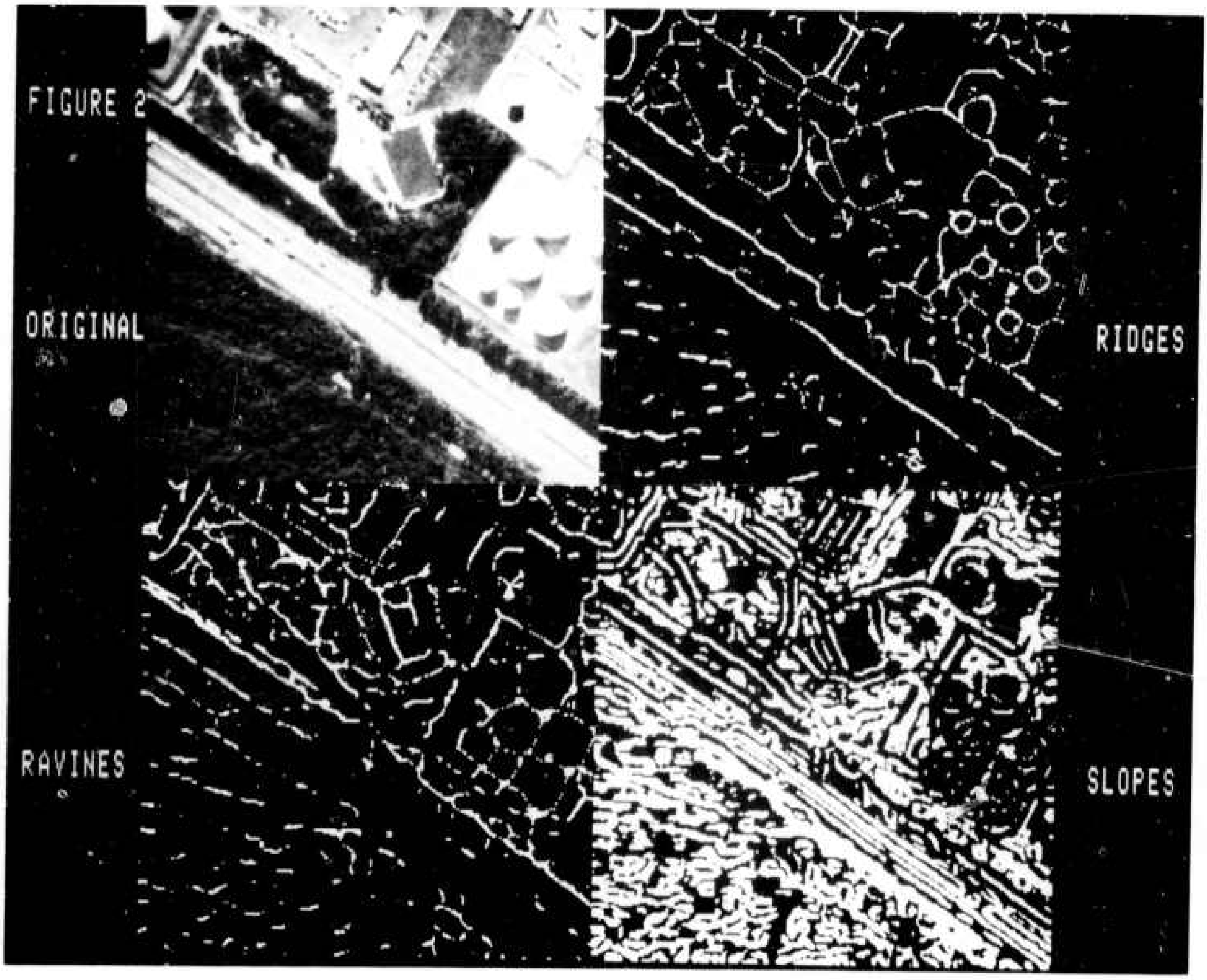


FIGURE 2
ORIGINAL
RIDGES
RAVINES
SLOPES

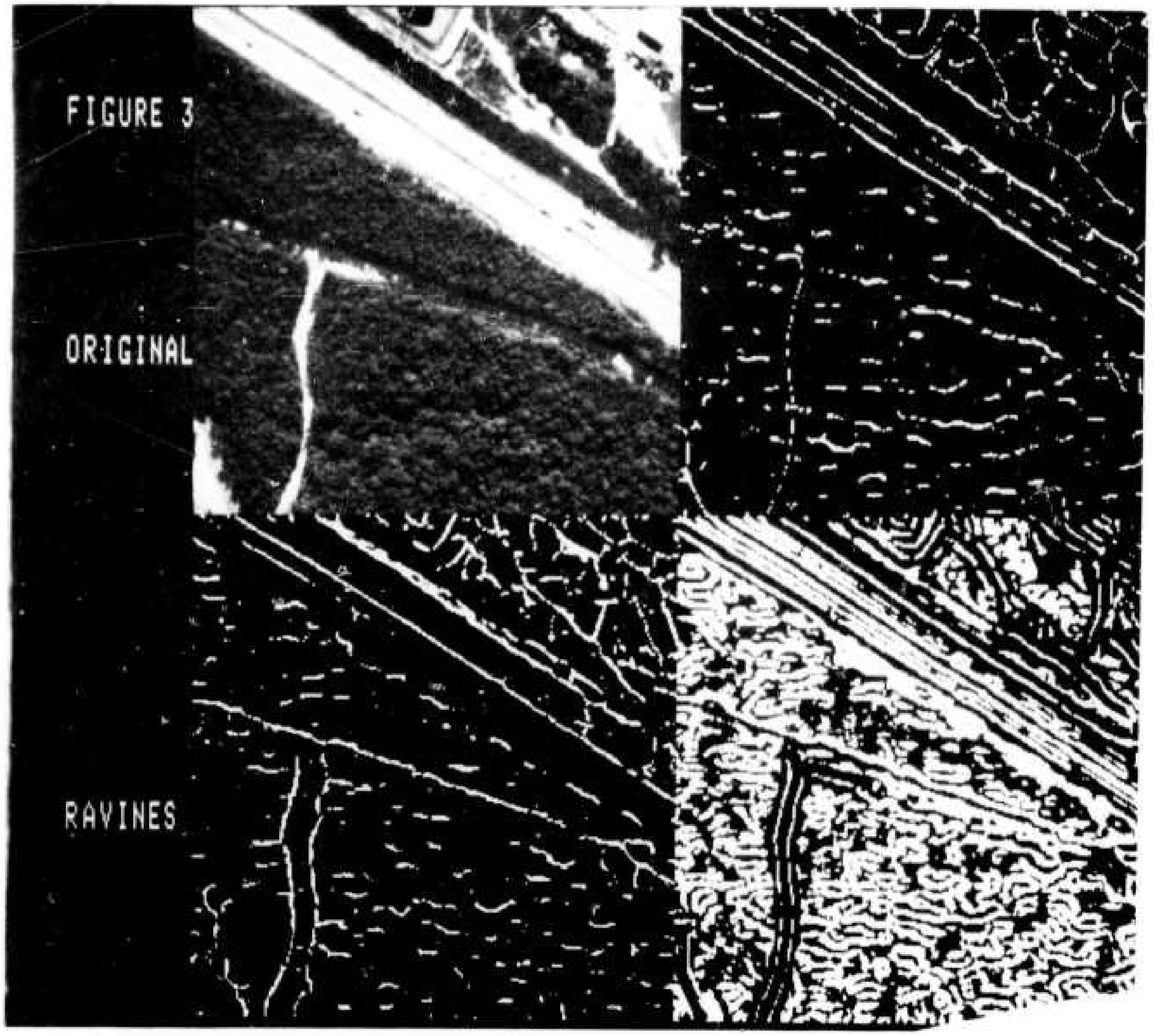


FIGURE 3
ORIGINAL
RAVINES

Monocular Reconstruction of a Complex Urban Scene in the 3D MOSAIC System

Martin Herman

Computer Science Department
Carnegie-Mellon University
Pittsburgh, PA 15213

Abstract

The author's

A system for obtaining a surface-based, three-dimensional description of a complex urban scene from a single image is described, and an example involving an aerial photograph is provided. Our approach exploits task-specific knowledge involving block-shaped objects in an urban scene. First, linear connected structures in the image are generated; these are meant to represent building boundaries. Next, the two-dimensional *2D structures are converted into* three-dimensional (3D) *3D wire frames. Finally, an approximate surface-based description of the scene is generated from the wire frames. The monocular analysis system is a component of the* 3D Mosaic *scene understanding system.*

1. Introduction

This paper describes a system for interpreting complex images of urban scenes. We show how a three-dimensional, surface-based description of such a scene is obtained from a single aerial photograph by exploiting task-specific knowledge involving block-shaped objects in an urban scene.

The work discussed here represents the monocular analysis component of the 3D Mosaic scene understanding system. The goal of this system is to automatically acquire a 3D description (or model) of a complex urban scene from multiple images.

In this paper, we first review briefly the 3D Mosaic system. Then the steps we perform during monocular analysis are discussed. These steps include extracting lines and junctions from the image, forming 2D linear structures, obtaining 3D wire frames from the 2D structures, and finally generating a surface-based description from the wire frames.

2. The 3D MOSAIC System

The 3D Mosaic system generates a scene model by incrementally accumulating information derived from multiple images obtained from multiple viewpoints. This approach arises from three observations: First, single images contain only partial information about a scene. Second, errors and inconsistencies often result when interpreting complex images. Third, features of visible surfaces (such as scene edges and texture) are often more apparent (and thus easier to extract) in one image than another because of differences in viewpoint and lighting conditions.

After each view of the scene is analyzed, it results in a partial 3D scene description which usually contains errors and omissions. The initial model, constructed from the 3D information obtained from the first view, represents an initial approximation of the scene. As each successive view is processed, the model is incrementally updated and gradually becomes more accurate and complete. At any point along its development, the model represents the current understanding of the scene and may be used for tasks such as matching, display generation, planning paths through the scene, and making other decisions dealing with the scene environment. Further details may be found in [3, 5, 4].

Each view of the scene may be either a single image or a stereo pair. Our previous reports have demonstrated how stereo analysis is used to obtain a scene model. In this paper, we will focus our discussion on the aerial photograph of Washington, D.C. shown in Fig. 1. In fact, our previous reports have shown how a stereo pair obtained from a different view of the same scene as in Fig. 1 has been processed to obtain a scene model. The result of processing the image in Fig. 1 should therefore be used to update the previously obtained scene model. The updating procedure, however, has been described elsewhere [3, 5, 4] and will not be discussed here. Instead, we will show how monocular analysis is used to obtain an approximate 3D reconstruction of the scene.

3. Monocular Analysis

Our approach to interpreting complex monocular images of urban scenes exploits task-specific knowledge. We assume that the objects in the scene are trihedral polyhedra containing only vertical and horizontal faces, i.e., faces perpendicular and parallel, respectively, to the ground plane. In addition, we assume that the vertical vanishing point in the image is given. Although finding vanishing points in images of outdoors scenes is a difficult problem, some progress has been made. For example, Barnard [1] describes a method for finding intersections of extended line segments in an image using Gaussian mapping.

The following steps are performed in our approach to monocular reconstruction. First, linear connected structures in the image that are meant to represent building boundaries are formed. The structures are obtained by first extracting junctions from the image, many of which arise from building corners. To obtain lines corresponding to building edges, we hypothesize connections between the junctions. The 2D structures are formed by linking the junctions and hypothesized lines.

These 2D structures are then converted into 3D wire frames using, in addition to the task-specific assumptions mentioned above, the following two general asumptions: lines in the image directed toward the vertical vanishing point are vertical in 3-space, and two lines that are aligned in the image are also aligned in 3-space.

Finally, task specific knowledge is used to convert the 3D wire-frame description into a surface-based description, or scene model. Examples of the scene model are shown in Figs. 15 and 16.

4. Extracting Lines and Junctions

The method we use for extracting lines and junctions is the same as that used during stereo analysis in the 3D Mosaic system [3, 5], where a junction-based stereo matching approach is used. The following is a brief review of this method.

The first step is to extract linear segments. A 3x3 Sobel operator is used to detect edge points. These are then thinned using a modified Nevatia and Babu algorithm [7], as shown in Fig. 2. The resulting edge points are linked and approximated by piecewise linear segments using the iterative end-point fitting algorithm [2, 7]. Finally, short lines are discarded. The resulting line image corresponding to Fig. 1 is shown in Fig. 3.

The next step is to extract junctions from the line image. A junction is a group of line segments (*legs*) in the image that meet at a point. We consider the following four junction types: L, ARROW, FORK, and T. To find junctions, a 5x5 window around each end point of each line is searched for ends of other lines. If the window contains the ends of three lines, they are classified as an ARROW, FORK, or T junction depending on the angles between the lines. If the window contains the ends of two lines, they are classified as an L junction. If the window contains more than three lines, each set of two lines is assumed to form a distinct L junction. Junctions that have been found in this manner are labeled in Fig. 3. Notice that many of the junctions correspond to building corners.

5. Locating 2D Structures

After lines and junctions are extracted, connected structures are formed by hypothesizing new lines to connect junctions. These lines are meant to correspond to building edges. Two steps are used in the process of hypothesizing connecting lines. First, two junctions may be connected only if a leg of one points at the other, i.e., the extended leg meets the other junction. Second, the two junctions must appear to be connected by line segments in the line image.

The first step involves finding all pairs of junctions such that one has a leg pointing at the other, and preceeds as follows. First, if two junctions share the same leg, they are connected. Next, for each leg of each junction J_i, a thin rectangular window is located in the direction along the leg (Fig. 4). Of the junctions within this window and within an angle α from the direction of the leg, the one closest to J_i is retained as a candidate for being connected to J_i. Fig. 5 shows a graph with all candidate connections drawn.

Only the connections in Fig. 5 that appear as connections in the line image (Fig. 3) are retained. The following procedure is used to determine this. For each pair of connected junctions J_i and J_k, we find all segments in the line image that are contained within a thin rectangular window connecting J_i and J_k (Fig. 6), and project these segments onto the line connecting the two junctions. Next we consider how much of this line is covered by projected segments. The connection between J_i and J_k is retained only if the percentage of coverage exceeds a threshold.

The result of this pruning step is shown in Fig. 7. Note that it does a good job in eliminating unwanted connections. We also tried another method to perform the pruning. It involved applying the Hough transform to a thin rectangular region in the edge image (Fig. 2) between each connected pair of junctions to determine whether a line connecting the junctions could be found. The results of this method were not as good, probably because the high number of texture edge points suggested too many lines.

At this point in the processing, junctions have two kinds of legs, those extracted in the junction finding step and those hypothesized as connections between junctions. Some of these legs are extraneous and are eliminated as follows. For each connected pair of junctions, if one has a leg that points at the other, the leg is deleted, for it is replaced by the hypothesized leg. Next, we utilize the assumptions that all vertices in the scene are trihedral and have one vertical and two horizontal legs, and that lines in the image directed toward the vertical vanishing point are vertical in the scene. First, each junction leg is labeled "vertical" or "horizontal" depending on whether or not it is directed toward the vertical vanishing point. Then, if a junction has more than one "vertical" or two "horizontal" legs, the extra ones are deleted according to a priority scheme that rates hypothesized legs higher than originally extracted ones, and rates hypothesized legs connecting two junctions that were originally mutually pointing higher than legs connecting junctions where only one was originally pointing at the other. The extra legs with the lowest priorities are deleted. The resulting legs are shown in Fig. 8. At this point, we have 2D connected structures representing portions of building boundaries.

6. Obtaining 3D Wire Frames

In order to obtain 3D information from the 2D structures in Fig. 8, we assume that the vertical vanishing point is known, that the camera focal length is known, that all lines part of the 2D structures (Fig. 8) arise from either vertical or horizontal scene edges, and that the lines can be labeled as such according to whether or not they are directed torward the vertical vanishing point. First, we calculate the vector from the focal point to the vertical vanishing point. This results in a 3-space vector in the vertical direction [1], which will be very useful for our processing.

Suppose we want to recover the 3D configuration of the junction $p_1p_2p_3p_4$ in Fig. 9 under the assumptions outlined above. Suppose also that line p_2p_4 has been labeled "vertical" and lines p_1p_2 and p_2p_3 have been laheled "horizontal". Let f be the focal length, and let $\boldsymbol{u}$ be the unit vector in the vertical direction. The vector $\boldsymbol{u}$ is normal to all horizontal planes. First we would like to determine the 3-space position of v_2, corresponding to the junction point p_2. Since it is impossible to determine the actual position of this point from a single image without special information, the position is determined as some arbitrary point lying on the ray through p_2. If the focal center is the origin of the coordinate system and $\vec{p_2}=(x_2',y_2',-f)$ and $\vec{v_2}=(x_2,y_2,z_2)$, then

$$\vec{v_2}=a\vec{p_2},$$

where a is the arbitrary distance from v_2 to the focal point.

The equation of the horizontal plane $v_1v_2v_3$ can now be established as

$$\vec{r}\cdot\boldsymbol{u}=\vec{v_2}\cdot\boldsymbol{u}$$

where $\boldsymbol{u}$ is the normal to the plane and $\vec{r}$ is any point contained by the plane. The 3-space positions of the points v_1 and v_3 can then be computed as the intersections of this plane with the rays through p_1 and p_3, respectively, i.e.,

$$\vec{v_1}=\frac{\vec{v_2}\cdot\boldsymbol{u}}{\vec{p_1}\cdot\boldsymbol{u}}\vec{p_1}$$

$$\vec{v_3}=\frac{\vec{v_2}\cdot\boldsymbol{u}}{\vec{p_3}\cdot\boldsymbol{u}}\vec{p_3}$$

Finally, the 3-space position of the point v_4 is computed as the intersection of the ray through p_4 with the line through v_2 along the vector $\boldsymbol{u}$:

$$\vec{v_4}=\frac{\|\vec{v_2}\times\boldsymbol{u}\|}{\|\vec{p_4}\times\boldsymbol{u}\|}\vec{p_4}.$$

Although this technique permits us to recover the 3D configuration of any junction relative to some arbitrary depth, it is not useful to apply it directly to the junctions in the original line image (Fig. 3) because the relative heights above the ground plane of the corresponding vertices cannot be determined; the height of each vertex is arbitrarily chosen without relation to the heights of other vertices. It is more useful, however, to apply the technique to the 2D structures in Fig. 8, since the heights of the vertices within each structure can be related. To see how this is done, consider the example in Fig. 10, which shows a 2D structure. The solid lines are part of the extracted structure, while the dashed lines are for the reader's convenience to make the 3D shape more apparent. Suppose lines p_1p_6 and p_3p_4 have heen labeled "vertical", while the other solid lines have been laheled "horizontal". Applying our technique to (say) point p_1, the 3-space positions of the vertices corresponding to points p_1, p_2, and p_6 can be determined relative to some arbitrary depth a for p_1. If the technique is applied next to point p_2, the 3-space position of point p_3 can be determined as a function of the depth a. This procedure continues with points p_6, p_4, and so on, until the 3D configuration of the whole structure has been determined, relative to some arbitrary depth.

In order to obtain a coherent scene description, the depths of the different structures in the scene must be related. We use two methods to do this. The first method involves finding structures that lie on the ground plane. Suppose a junction point p of such a structure is hypothesized to arise from a vertex lying on the ground. Then the 3-space position $\vec{v}$ of the vertex is

$$\vec{v}=\frac{d}{\vec{p}\cdot\boldsymbol{u}}\vec{p},$$

where $\boldsymbol{u}$ is the unit vector in the vertical direction (thus normal to the ground plane) and d is an arhitrarily chosen distance from the origin to the ground plane. Since the 3-space position of all junctions arising from ground points can be calculated in this manner, the depths of all structures containing such points can be related to one another through the parameter d.

To hypothesize junctions in the 2D structures (Fig. 8) that arise from vertices lying on the ground plane, we use the observation that if a line labeled "vertical" connects two junctions, the line is directed toward the vertical vanishing point with respect to one junction, but away from this vanishing point with respect to the other junction. The latter junction is assumed to represent a vertex lying on the ground plane. Points p_1 and p_3 in Fig. 10 are examples of such junctions. The 3-space positions of these junctions are then calculated, and their values are propagated throughout their structures as described previously. Fig. 11 depicts a perspective view of the 3D wire frames obtained in this manner.

There are many structures in Fig. 8 that do not contain points lying on the ground plane, either because such points are occluded in the scene or because they have not been properly extracted from the image. Neverthless, the heights of some of these structures can be hypothesized using the rule that if two lines are aligned in the image, assume they are also aligned in 3-space [6]. To see how this rule is used, consider Fig. 12. Suppose that points p_1 through p_7 have already been assigned 3D coordinates, and we want to obtain the 3-space position of the 2D structure $p_8p_9p_{10}p_{11}$. Since the lines p_6p_7 and p_8p_{11} are aligned in the image and both are labeled "horizontal", they are assumed to be aligned in the scene and to lie in the same horizontal plane. The equation of this plane is

$$\vec{r}\cdot \boldsymbol{u} = \vec{v_6}\cdot \boldsymbol{u},$$

where v_6 is the 3-space point corresponding to p_6, and $\boldsymbol{u}$ is the unit vector in the vertical direction. The 3-space position of the point p_8 is therefore determined as the intersection of this plane with the ray through p_8, or

$$\vec{v_6} = \frac{\vec{v_6}\cdot \boldsymbol{u}}{\vec{p_8}\cdot \boldsymbol{u}}\,\vec{p_8}.$$

The 3D coordinates of this point may then be propagated to points p_9, p_{10}, and p_{11} as described preveously. Note that all 3D positions are functions of the parameter d, which is arbitrarily chosen for the equation of the ground plane.

The following tests are used to determine whether two lines in the image, $l1$ and $l2$, are aligned (Fig. 13):

1. They must be almost parallel, i.e., the smallest angle between them must be less than the threshold angle β.

2. They cannot be displaced laterally by too much, i.e., $l2$ must lie totally within a band of threshold thickness W surrounding $l1$.

3. They cannot be too far from each other, i.e., the ratio of the gap g between the two lines to the average length of the lines must be less than a threshold. Although this condition is not required for strict alignment, we include it so that the alignment rule will not be applied to two lines whose distance from one another is large compared to their lengths.

4. They must be separated from each other, i.e., the projection of $l2$ onto $l1$ cannot overlap $l1$. Again, this condition, although not required for strict alignment, is included so as not to apply the alignment rule to two lines that are not separated enough.

Fig. 14 depicts a perspective view of the final 3D wire frames obtained using both the methods of hypothesizing points on the ground plane and applying the alignment rule.

7. Generating the 3D Scene Model

In order to obtain a 3D scene model, we use a geometric modelling component that converts a wire-frame description of a scene into a surface-based description. This geometric modelling component is also used during stereo reconstruction in the 3D Mosaic system, since the output of the stereo analysis is a wire-frame description [3, 5, 4]. The following is a brief review of the geometric modelling system.

To generate the surface-based description from the wire frames, it is assumed that the objects in the scene can be approximated by polyhedra, that each face is a parallelogram unless there is contrary evidence, that the position of the ground plane is known, and that the objects have walls that are perpendicular to the ground plane. (It is not assumed that the roofs are parallel to the ground.)

The processing proceeds as follows. First, wire-frame edges that are nearly parallel and very close to each other are merged. Next, *web faces* that correspond to corners of planar faces are generated for each vertex corner. The web faces that represent corners of a single face are then merged. After all mergers, faces that do not have closed boundaries are completed either as parallelograms or as other closed polygons. After all faces are completed, those that seem to be holes in other faces are converted to holes. Finally, objects that are not closed are completed by dropping vertical walls from the roofs toward the ground plane.

Perspective views of the resulting scene model are shown in Fig. 15. In order to render more realistic displays, gray scale obtained from the original image (Fig. 1) is added to them (Fig. 16).

8. Conclusion

A system has been described that reconstructs the three-dimensional shape of a complex urban scene from a single image. A 3D wire-frame description of the scene, meant to represent portions of building boundaries, is generated first. The wire frames are then converted into an approximate surface-based 3D model of the scene. We have also demonstrated that task-specific knowledge is very useful for interpreting complex images. Such knowledge is used for generating 2D image structures, 3D wire frames, and the 3D scene model.

There are several extensions and improvements we have in mind for the methods and techniques described in this paper:

1. The 2D structures are obtained by hypothesizing line

connections between extracted junctions. More complete 2D structures might be obtained by hypothesizing and testing for lines elsewhere in the image. Task-specific knowledge can be useful for hypothesizing such lines.

2. There are many sources of knowledge, such as shadows [8] and texture, which should be incorporated into our monocular analysis.

3. The junction-based stereo matching used in the 3D Mosaic system might be improved if it were modified to match the kinds of 2D image structures described in this paper.

Acknowledgement

Takeo Kanade and Yu-ichi Ohta have provided useful comments and discussions throughout the development and implementation of the ideas in this paper.

References

1. Barnard, S. T. "Methods for Interpreting Perspective Images." *Proc. ARPA Image Understanding Workshop* (September 1982).

2. Duda, R.O. and Hart, P. E.. *Pattern Classification and Scene Analysis.* John Wiley and Sons, New York, 1973.

3. Herman, M., Kanade, T., and Kuroe, S. "Incremental Acquisition of a Three-Dimensional Scene Model from Images." *Proc. ARPA Image Understanding Workshop* (September 1982).

4. Herman, M., Kanade, T., and Kuroe, S. "The 3D MOSAIC Scene Understanding System." *Proc. IJCAI-83* (August 1983).

5. Herman, M. and Kanade, T. Incremental Acquisition of a Three-Dimensional Scene Description from Complex Images. Computer Science Department, Carnegie-Mellon University, in preparation.

6. Lowe, D. G., and Binford, T. O. "The Interpretation of Three-dimensional Structure from Image Curves." *Proc. IJCAI-81* (August 1981).

7. Nevatia, R. and Babu, K.R. "Linear Feature Extraction and Description." *Computer Graphics and Image Processing 13* (July 1980), 257-269.

8. Shafer, S. A., and Kanade, T. Using Shadows in Finding Surface Orientations. Tech. Rept. CMU-CS-82-100, Carnegie-Mellon University, January, 1982.

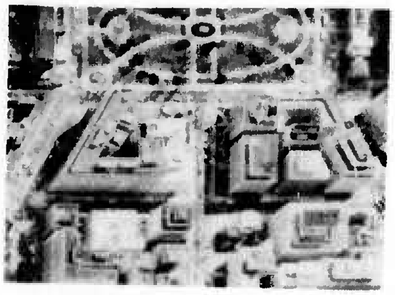


Figure 1: Aerial photograph showing part of Washington, D.C.

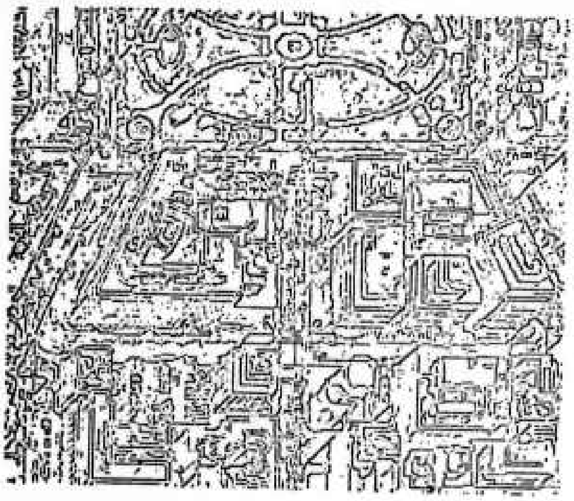


Figure 2: Result of thinning the edges obtained by applying a Sobel operator to the image of Fig. 1.

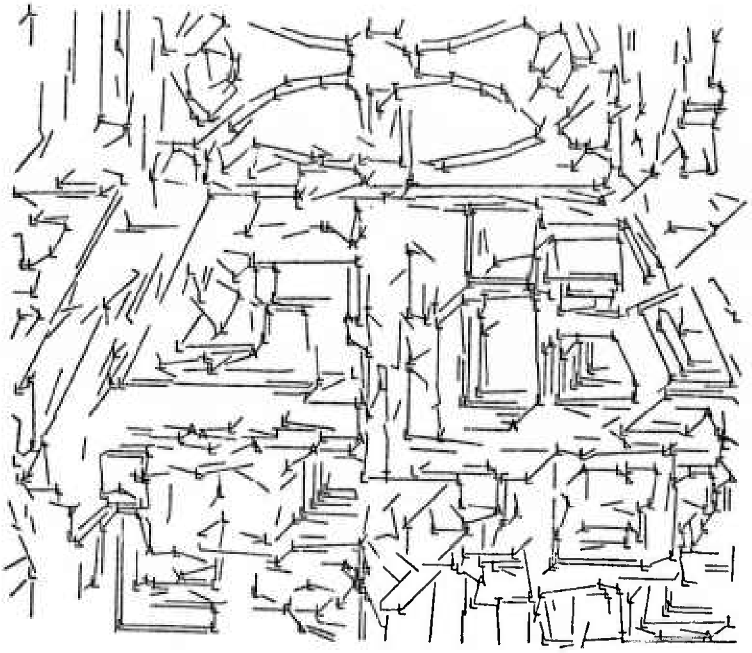


Figure 3: Lines fitted to the edge points of Fig. 2 after they are linked. Junctions in the image are classified as L, A (arrow), F (fork), or T.

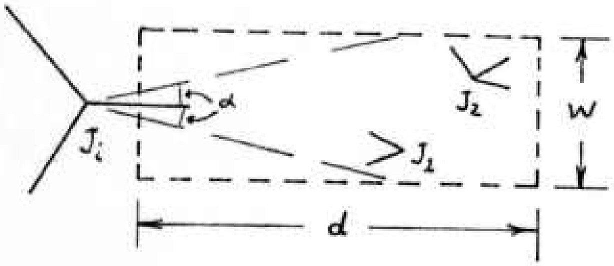


Figure 4: The closest junction to J_i within the thin rectangular window of length d and height W, and within the angle 2α, is a candidate for being connected to J_i.

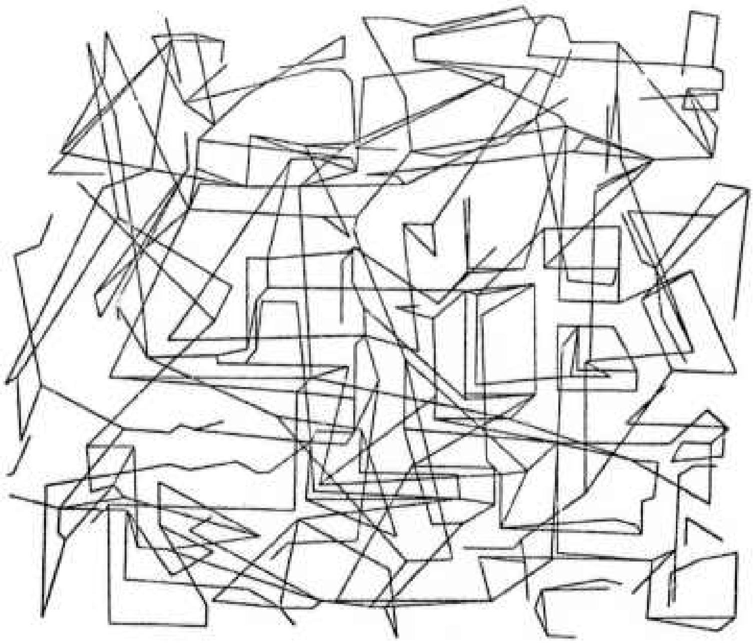


Figure 5: Each line represents a possible connection between the junctions at its two end points. Each end point corresponds to a junction in Fig. 3.

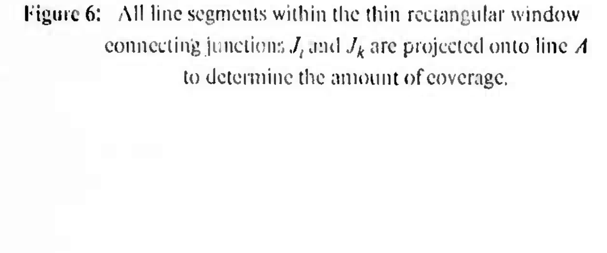


Figure 6: All line segments within the thin rectangular window connecting junctions J_i and J_k are projected onto line A to determine the amount of coverage.

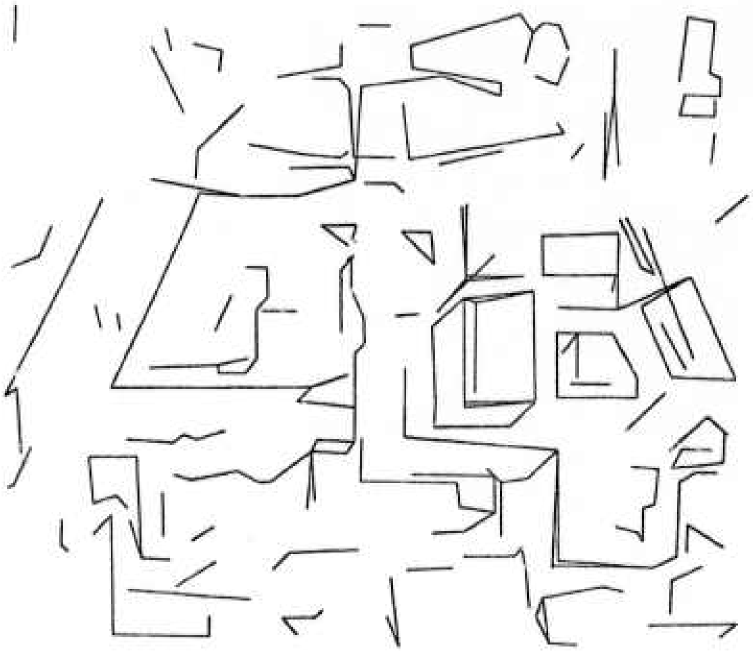


Figure 7: Result of pruning the junction connections in Fig. 5 by determining whether segments in Fig. 3 adequately cover the area between each pair of connected junctions.

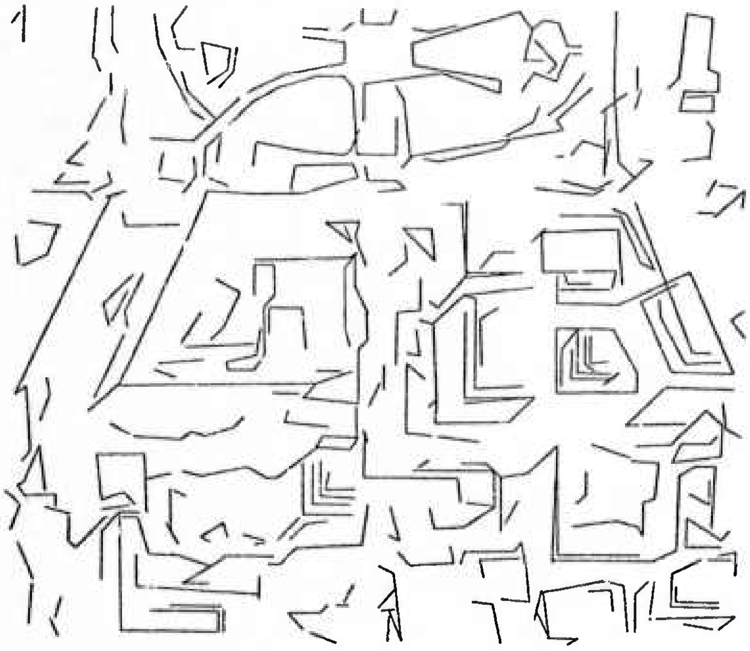


Figure 8: Result of adding to Fig. 7 the junction legs that were originally extracted in the junction finding step, and then deleting extraneous legs.

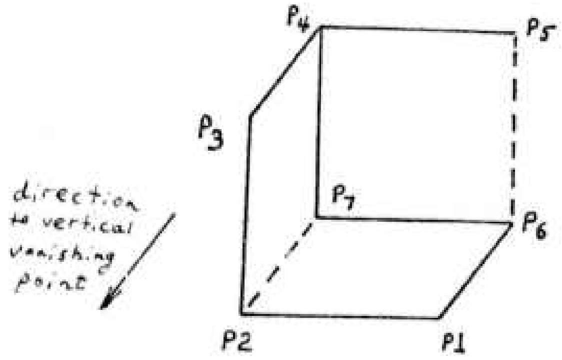


Figure 10: The solid lines represent a connected 2D structure. The dashed lines are for the reader s convenience to make the 3D shape more apparent.

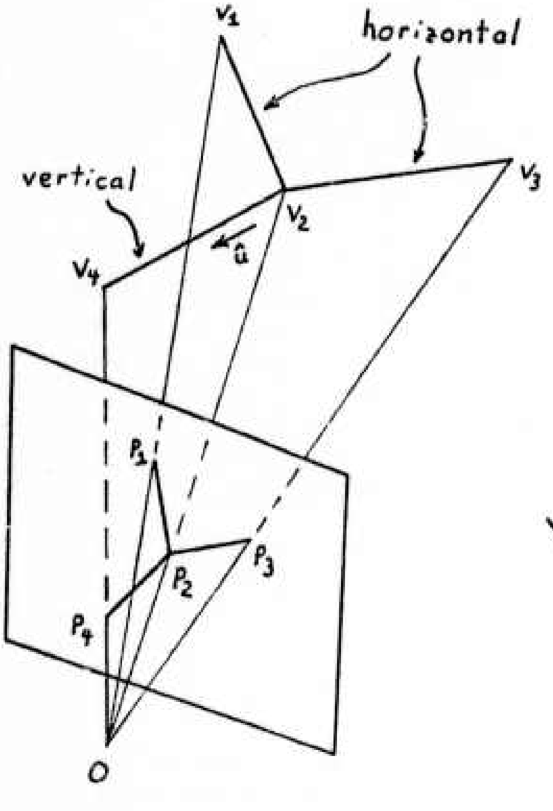


Figure 9: The 3D configuration of the junction $p_1p_2p_3p_4$ can be recovered under assumptions explained in the text. O is the focal center and the origin of the coordinate system.

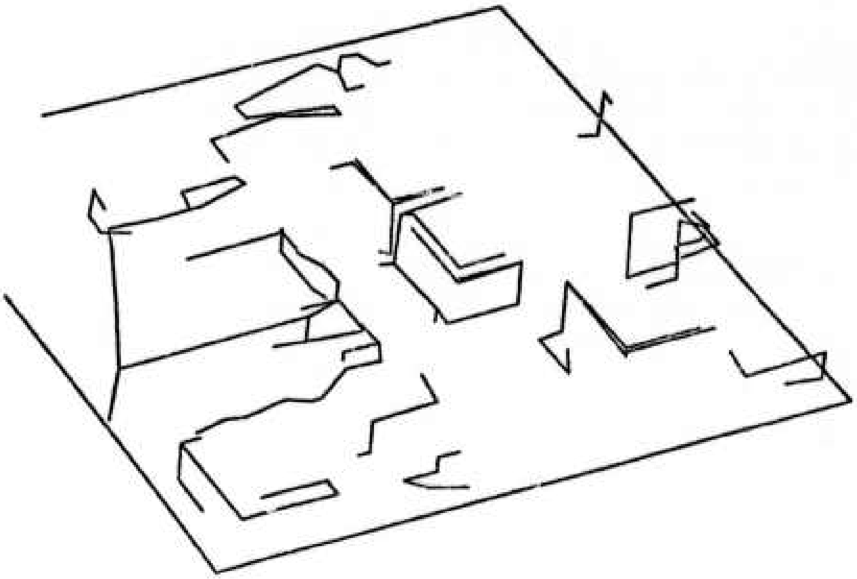


Figure 11: Perspective view of 3D wire frames generated from Fig. 8 using the method of finding junctions arising from vertices lying on the ground plane.

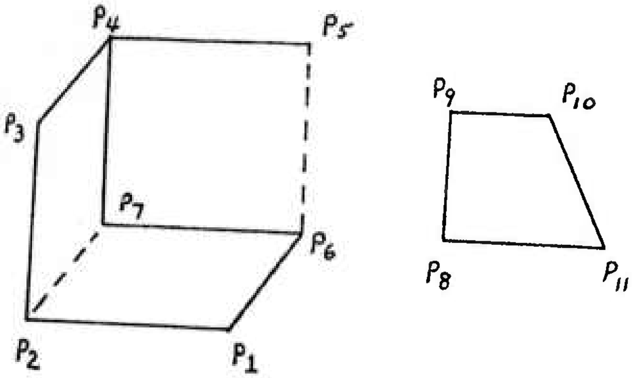


Figure 12: If the 3D configuration of the structure on the left has been determined, the relative 3D position of the structure on the right may also be determined because lines p_6p_7 and p_8p_{11} are aligned.

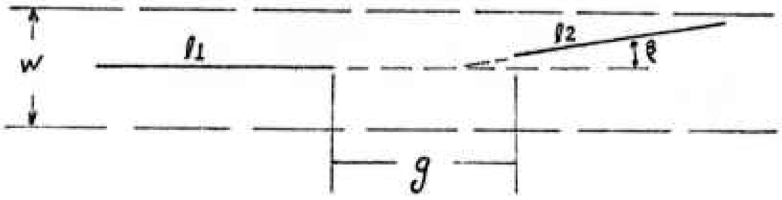


Figure 13: The lines $l1$ and $l2$ are aligned.

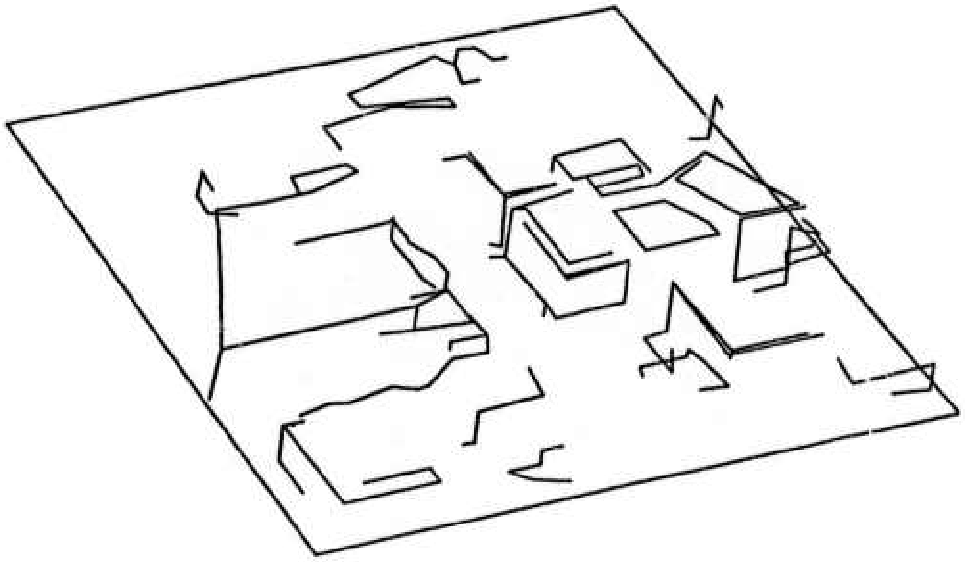


Figure 14: Perspective view of final set of 3D wire frames generated from Fig. 3.

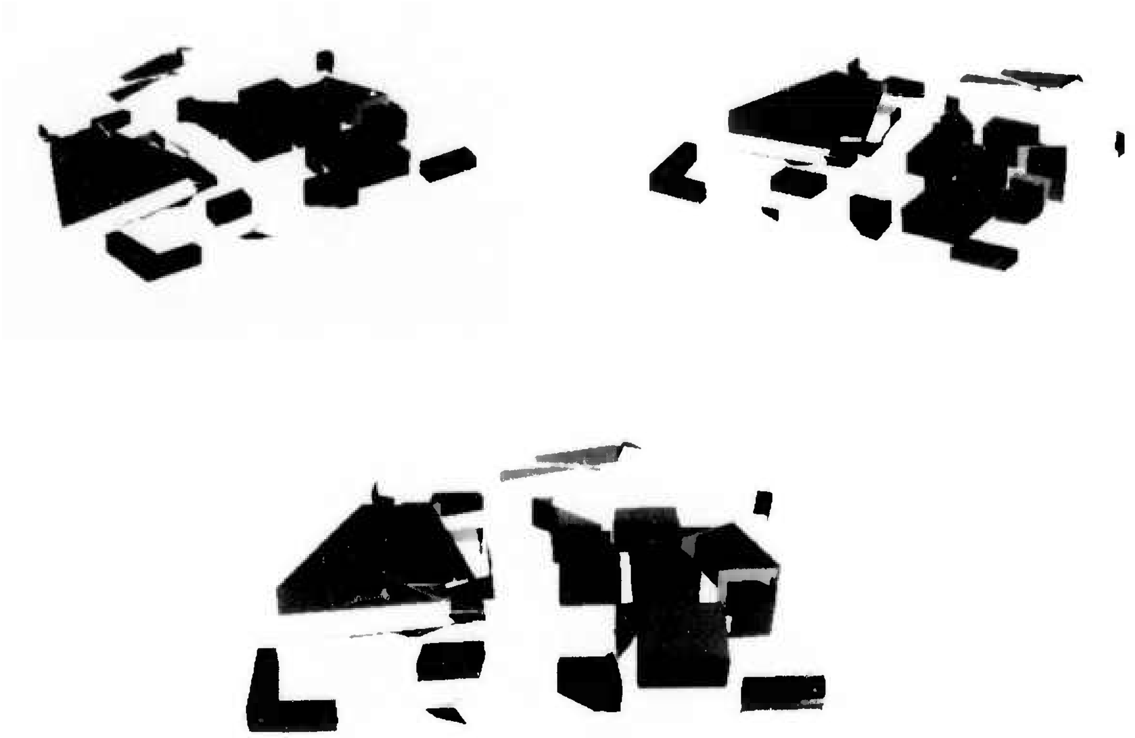


Figure 15: Perspective views of reconstructed buildings.

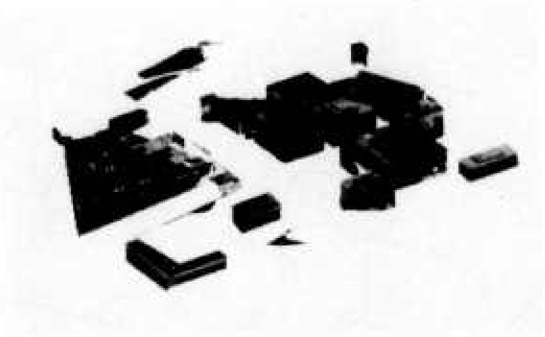


Figure 16: Reconstructed buildings of Fig. 15 with gray scale derived from Fig. 1 mapped onto faces

PROGRESS IN STEREO MAPPING

H. Harlyn Baker, Thomas O. Binford, Jitendra Malik, Jean-Frederic Meller
Artificial Intelligence Laboratory, Computer Science Department,
Stanford University, Stanford 94305.

The authors

1: Introduction

Rather than a description of a single piece of research, this is more in the line of a collective report on some areas we've been addressing in our research in stereo mapping. We have been developing tools and experimenting with matching strategies that will build to a rule-based stereo matching system. In particular, we have been:

they

a) demonstrating the design and the utility of the rule-based approach to surface inference from monocular information through the hand synthesis of matching strategies;

b) developing tools to support a mapping interactive test facility;

c) experimenting with the [Baker 1981] stereo mapping system, preparing to run it on some new imagery.

In a), we have been carrying out research aimed at the analysis and synthesis of rules for inference of three-dimensional shape from single images. We have been addressing inference of matching rules and the use of model-based analysis both with theoretical analyses and with hand and automated analyses of specific matching strategies; the latter applied to both real and synthesized imagery examples. This inference also has application in constraining search for matches in stereo correspondence.

Our work in developing tools has centered on:

a) a system for the hand construction of edge descriptions from hard copy imagery;

b) an interactive system for determining the transform to bring image pairs into collinear epipolar registration.

Both of these tools make extensive use of interactive graphics, and the latter takes much advantage of previous stereo research from our laboratory ([Gennery 1980]).

In c), we have been undertaking to apply the [Baker 1981] system to some new imagery. In this we hope to demonstrate its effectiveness, to expose its limitations, and to suggest both its role in an advanced mapping system and complementary research needed to improve its utility. Significant restructuring of the system was called for in enabling it to process this new imagery. Details of these changes are described in section 4, which deals with the matching process. Modifications have now been implemented, enabling the system to:

- function on the output of an improved edge operator [Marimont 1982];
- use *edge extent* as one of its parameters in seeking optimized correspondence;
- exploit prepared transform information in processing images whose epipolar lines are not collinear with the scanning axes of the cameras.

We will describe results in these areas of the research through discussion of the following:

a) the use of image edge descriptions produced using the digitizing facility and from an automated process [Marimont 1982] in synthesizing rules for stereo matching;

b) development of a system for epipolar registration of image pairs;

c) modifications to the [Baker 1981] stereo system (results later).

2: Inference and Modelling

2.1 Preamble – the digitizing facility

Our approach to rule development begins with hand synthesized and some automatically generated edge data. We have systems for the automatic generation of edge data (*i.e.* [Marimont 1982]). This data, extracted from a sufficiently wide selection of imagery types, gives good insight into the current capabilities of automated processes. Automated processes, however, are not able presently to give as meaningful a description of an image as we would like, and have not been designed to provide the aggregrated abstractions research systems ([Lowe 1982]) will be soon supplying. To bridge this inadequacy, we work with both automatically generated data (the current state-of-the-art), and hand generated data (representative of the next generation of edge analysis processes). The hand generated data is obtained from a manually operated digitizing tablet. We have written a graphics-based digitizing and editing system to run with a GTCO tablet in producing these image descriptions.

Figure 2-1 below shows an image pair of a building complex (referred to as the Sacramento imagery). Figure 2-2 shows the results of tablet edge extraction on these images. Figure 2-3 shows the results of the Marimont operator [Marimont 1982] on the image pair. Manually generated edge data was produced using this facility for the analysis of rule synthesis of section 2. It was also used to digitize the building data of figures 2-1 for input to the OTV inference process, as figure 2-5.

2.2 Data for Rule Synthesis

We have obtained extended edge data from hand and automated processing for use in synthesis of matching rules. Results from earlier work on OTV analysis (orthogonal trihedral vertices) have been exploited [Perkins 1968] for rule formation in shape inference and in constraining search for correspondence. We have taken examples from the modelling of generic structures to prodoce ground and aerial views of a building complex, and have used this, as well as other data, in rule synthesis.

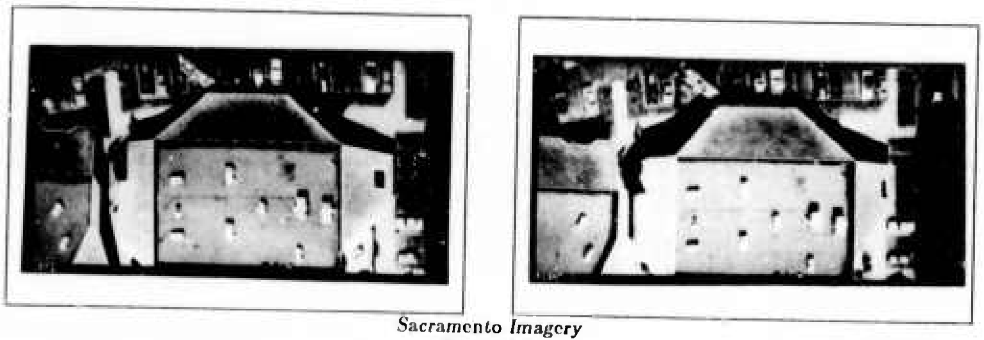
2.3 Modelling and Vision

2.3.1 – Modelling, prediction and interpretation

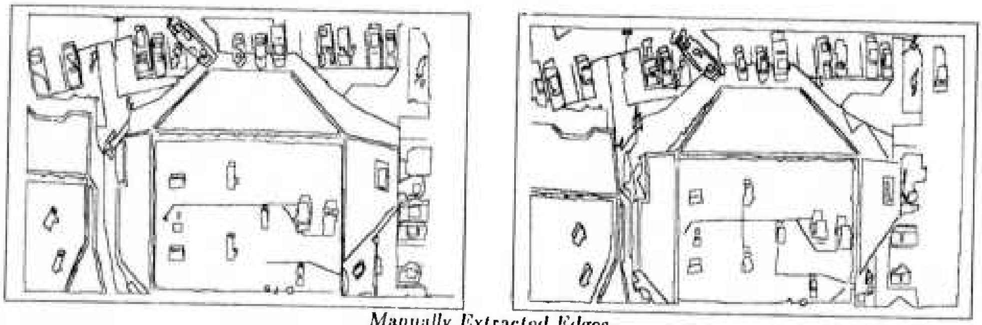
Of course, one of the primary goals of research in computer vision is the development of systems that can recognize and locate objects in images. In order to identify such an object, it is clearly necessary to have some description of its characteristics that can be detected in an image. A *representation* of an object is the form this description takes.

One approach to representation is to provide the system with three-dimensional models of objects. Rotation of these models will allow objects to be observed, conceptually, from differing viewpoints. If parameters in a particular model are allowed to vary it is possible to have that single model represent a whole class of objects; constraining the parameters functions to delimit sub-classes. Further model manipulations, such as partitioning and projection, can be used to aid in mapping model to imagery data. The information contained in such object models may be used to determine possible interpretations of image features (*e.g.* edges, ribbons, corners) and to provide feedback to predict the locations of such features in an image.

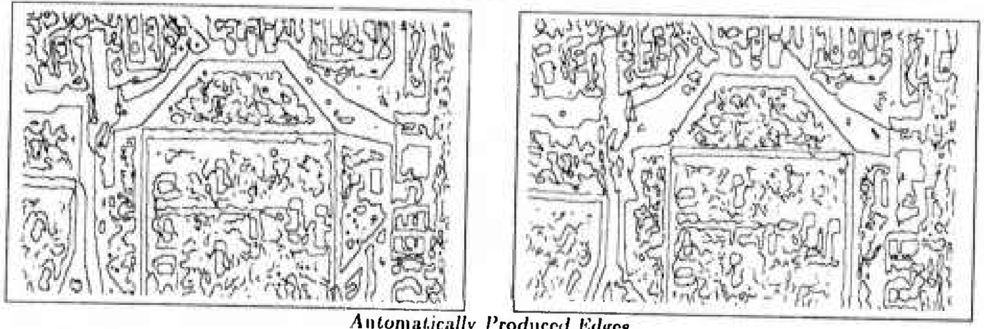
ACRONYM [Brooks 1981] is a three-dimensional rule-based modelling/vision system developed here at Stanford that provides, among other things, such feature prediction, model manipulation, and image inter-



Sacramento Imagery
Figure 2-1



Manually Extracted Edges
Figure 2-2



Automatically Produced Edges
Figure 2-3

pretation. The rule-base operates on the models and on the sensed data to accomplish scene interpretation. Such a rule-based approach has been shown to be an effective form for constraint and search implementation, and allows easy modification and addition of new rules without the need of altering the underlying code.

Our group's intention over the next few years is to build a rule-based stereo system operating within ACRONYM whose functioning will include model-based prediction. Working toward this, we have been carrying out experiments on scene inference and model-based prediction that will lead to a repertoire of stereo matching rules.

2.3.2 – Models and stereo matching

One of the major difficulties in determining stereo correspondence is in dealing with the large number of matches that are possible. Solution is generally found by search through a large parameter space, where possible correspondences are limited by geometric or photometric constraints. Search can be reduced even more dramatically by endowing the matcher with broad domain specific and domain independent knowledge. Such knowledge can be rule-based and model-based. Our proposition here is that the three-dimensional information in object models, along with inference and prediction mechanisms, can be used to interpret features in image pairs. These interpretations can then be used as filters to constrain the matching. We demonstrate this notion with the example of Orthogonal Trihedral Vertices, often referred to as cube corners or OTVs. Other rules synthesized from analysis of both manually extracted and automated edge processes follow.

The work on cube corners points to additional usefulness for a model-based approach. OTV orientation analysis (from matches across pairs of views) yields almost complete solution for camera parmameters; constraints on sizes (again, from rules and models) could complete the camera solution. But the orientation information yielded by a match of a pair of vertices is valid only if the vertex is a cube corner; thus it is necessary to be able to distinguish between vertices that are cube corners and those that are not. If the models contain sufficient information to identify cube corners, then the problem of determining cube corners independently of the identification process is eliminated. In fact, both the search for cube corners and the search for identification are likely to be reduced when they are combined.

2.3.3 – OTV rule-based analysis

OTV theory

In cultural scenes, we find a large number of interior and exterior corners of cubes – typically when two walls at right angles meet the roof or the floor. The importance of utilizing this common structural element – the Orthogonal Trihedral Vertex (OTV) – has been emphasized earlier[Liebes 1981]. Since they provide a very tight constraint – the three edges are mutually orthogonal in space – it is possible to calculate the three dimensional orientations given the projections in the image. This can be done for both orthographic and perspective viewing.

If the eye is assumed to be focussed on the vertex of the cube corner, perspective can be ignored and the projection of a cube corner in XYZ space will simply be its orthogonal projection on the XY plane.

Suppose that some 3 - star has angles between its rays a, b and c and also that the rays are represented by the unit vectors v_1, v_2, v_3. We are interested in detecting whether there are three vectors in XYZ space, which are mutually orthogonal and project, respectively to v_1, v_2, v_3.

Since projection is accomplished by dropping the z component, any 3 such vectors must be of the form $v_1 + \lambda_1 z$, $v_2 + \lambda_2 z$, and $v_3 + \lambda_3 z$ where z is the unit vector in the z direction.

Requiring mutual orthogonality implies that the dot products of these vectors in pairs be zero. From these conditiuns and some simple manipulations we can calculate the formulas for

$$\lambda_1 = \pm\sqrt{-\frac{(\cos a)(\cos c)}{(\cos b)}}, \quad \lambda_2 = \pm\sqrt{-\frac{(\cos a)(\cos b)}{(\cos c)}}, \quad \lambda_3 = \pm\sqrt{-\frac{(\cos c)(\cos b)}{(\cos a)}}$$

Hence solutions exist if

a) $\cos a$, $\cos b$, $\cos c$ are all non-zero and

b) either one or three of $\cos a$, $\cos b$, $\cos c$ are negative, so that the quantities under the square root sign are positive.

These results were first derived in [Perkins 1968].

Thus we have a way of both eliminating false candidates for being OTVs and finding the 3-D orientations of valid OTV's. This algorithm has been implemented and run on data from the digitizing tablet.

OTV with/from models

Our analysis begins on both images, processing bottom up on the two images separately. As the rule system identifies likely OTVs in images (from its models), it proceeds to match them. The system should already have a tentative identification of the buildings containing the OTVs, so there should be relatively few possible matches at this point. Only OTVs that could be the same point on the same object need be compared. The analysis results in depths of matched objects, for all those objects having OTVs.

This requires that the modelling system handle point elements, and that it include both:

- inferring OTVs from models (volumes);
- accessing OTVs stored explicitly with the models.

2.4 Rule Synthesis

2.4.1 – Inference rules

We continue with the development of inference rules. This work is a logical extension of previous work [Binford 1981, Lowe 1982] done at Stanford in developing rules for inferring surface information from a single view. General assumptions about illumination, object geometry, the imaging process etc. have been used to derive rules for making specific inferences. For stereo vision Arnold and Binford [Arnold 1980] have developed conditions on correspondence of edge and surface intervals. We divide our rules into two categories: monocular rules, which enable surface inference from a single view; and stereo rules, which facilitate cross-image matching.

2.4.2 - Monocular and stereo rules.

1. Monocular rules – Rules which have been developed for inferences from monocular views can be utilised to provide a partial 3-dimensional interpretation which directs search in the second view. This category includes the rule for interpretation of Orthogonal Trihedral Vertices.

   Another example is the T-junction rule [Binford 1981] which states that *'In absence of evidence to the contrary, the stem of a T is not nearer than the top, i.e. is coincident in space or further away'*. Application of this rule gives a set of nearer/farther relations. A hypothesized correspondence of edges which leads to inconsistent conclusions from the two views can be pruned from the search.

   An image line which is straight must be the image of a straight space curve unless the curve is planar and the observer is coincidentally aligned with the plane of curvature. This enables us to dismiss correspondences between straight edges in one view and curves in the other view. If two image curves are projectively consistent with parallel, we assume they are images of curves which are parallel in space. That implies that their images in the other view would be parallel *i.e.* parallels map to parallels.

   As these examples illustrate, most of the rules in [Binford 1981, Lowe 1982] and others developed by Malik and Binford have as direct corollaries stereo rules for checking the legality of a match. They can even direct the search process.

2. Stereo Rules – these are rules which have been derived from the stereo imaging process, and are a function of the imaging geometry.

An example rule in this class, which has long been used for finding stereo correspondences, is the epipolars rule – corresponding points must lie on corresponding epipolar lines. These rules have inherently no monocular analogs. Here are a few:

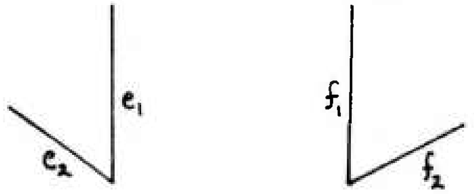
a) Horizontal planes in one view get mapped to horizontal planes in the other view.

b) Use of projective and quasi-projective invariants. This has not been examined in detail. Duda and Hart[Duda 1973] devote a chapter to this topic which has not really been exploited in stereo work.

c) Conditions on correspondence of edges and surface intervals[Arnold 1980].

d) Surface Occlusion rules:

   Surfaces visible in one view can be occluded in the other view. We are interested in the conditiuns when this takes place. The basic idea is that if we cross a surface, an obscuration of edge occurs. A left surface visible in a right view is visible in the left view unless there is obscuration by a tall object. Similarly a right surface visible in a left view will be seen unless obscured by a tall object. These surface-obscuration rules can be formalized by the cross-product rule:

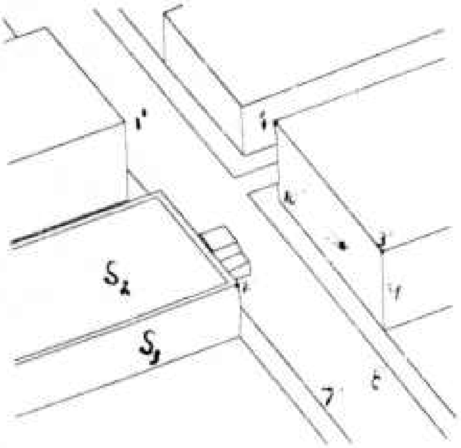


   For the hypothesized edge match e_1 with f_1 and e_2 with f_2, we compute the Z-component of the vector cross-product in the left image pair and the right image pair. If the z-components have opposite signs, we are seeing opposite sides of the surface. That implies that the object is not opaque.

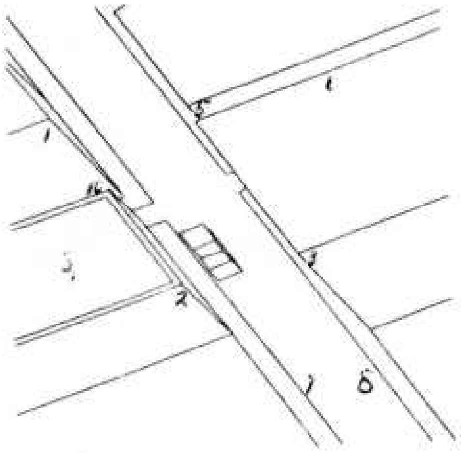
2.4.3 – Use of inference rules in a test analysis

Our preliminary results indicate good potential for the success of this approach. On hand simulations with line drawings of stereo pairs, the rules helped narrow down the choices considerably.

Consider the imagery shown in figures 2-4 and 2-5. Figure 2-5 is the right view and 2-4 the left view. Vertices 1, 2, 3 are orthogonal trihedral vertices. Using the formulae developed earlier, we can find the 3-D orientations of the edge vectors. These can be matched with the 3-D orientations of 1′, 2′, 3′ to obtain a registering of these vertices when combined with the epipolar constraint. All OTV's in one view need not be visible in the other *i.e.* 4′. Of the monocular constraints, the other major constraints which can be seen here are the T-junction rule and the parallels rule. In figure 2-5 edge 5 is behind edge 6. Edges 7 and 8 are parallel and so are 7′ and 8′. A match of 8 with 9′ would not be accepted. Surfaces S_1 and S_2 are both horizontal planes (as can be deduced from the OTV analysis) and can be matched. Surface S_3 is not horizontal. Here of course, this does not provide any new information. Surface S_4 being a left face in a left view is not guaranteed to be visible in the other view - as in fact it is. The cross-product rule could be used to dismiss a match between 10 and 10′.



Description of Left Image of Stereo Pair
Figure 2-4



Description of Right Image of Stereo Pair
Figure 2-5

3: Image Registration

3.1 Introduction to Epipolar Geometry

The search process in automated stereo mapping can be greatly restricted, and computation times significantly reduced, if information is available relating the relative camera geometries of a stereo pair. Often this information is available in the reconnaissance data (or at least a rough approximation to it). Other manual and automated schemes have been devised to provide the information when it is not present with the imagery (see [Hallert 1960], [Gennery 1980]). This camera geometry information allows establishing epipolar correspondence of lines across images. When this has been done, search in one image for match points of a feature in the other image can be constrained along a single vector. More generally, any features lying along a particular vector in the one image may be found along a single vector in the other image. These image plane vectors are termed epipolar lines. Corresponding vectors are termed corresponding or conjugate epipolar lines.

In this section we detail an algorithm for determining conjugate epipolar lines in a set of imagery for which such camera geometry information is not explicitly available. Here, we rely upon an operator to select corresponding points in the two images. The system automatically improves the resolution of the correspondence through Fourier interpolation over a match window [Gennery 1980]. The set of such points is taken by an automated camera solver to produce the needed geometric information. This point selection is done with pan/zoom cursor control on a graphics device. If the camera information is available (as, for example, from reconnaissance data), then the point matching phase may be omitted (although this provision has not been enabled in the current system). Equally, rough camera geometry information, if available, may be used to partially automate the point selection phase, although again this is not implemented here. [Gennery 1980] and [Moravec 1980] have implemented totally automatic camera solvers in their stereo matching systems. Our next improvement to this registration system will be to incorporate the image sampling and feature matching of the [Gennery 1980] system, removing the need for manual point selection.

3.2 Glossary of Terms

Given two cameras C_1 and C_2 with origins θ_1 and θ_2 and focal planes P_1 and P_2, we call:

Bundle of lines: A family of lines containing a common point.

Epipolar coordinates of a point: The number of the epipolar line it belongs to, and its distance to a fixed reference, like the epipole if it exists.

Epipolar direction: The direction of all epipolar lines if they are parallel.

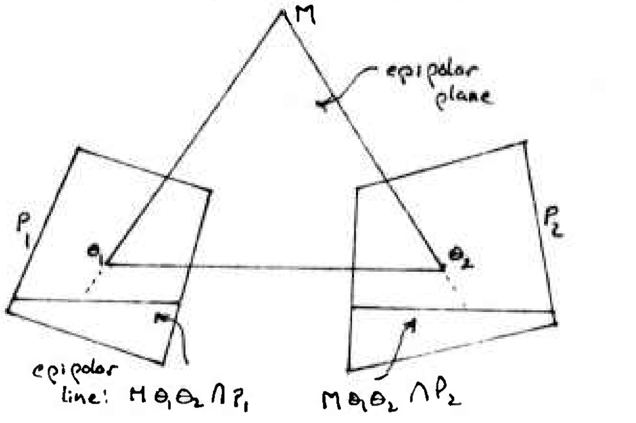
Epipolar line: The intersection of an epipolar plane with a focal plane. Alternate definition: the image in one camera of the pre-image of a point in the other camera's focal plane.

Epipolar plane: Any plane containing the two camera centers θ_1 and θ_2.

Epipolar space: A space where the coordinates are the epipolar coordinates. In this space, a horizontal line is an epipolar line, and the epipole, if it exists, is a whole vertical line.

Epipole: The intersection, if it exists, of all epipolar lines in a focal plane.

Conjugate epipolar lines: the intersections of an epipolar plane with the two focal planes.


M
epipolar plane
P_1
P_2
θ_1
θ_2
epipolar line: $M\theta_1\theta_2 \cap P_1$
$M\theta_1\theta_2 \cap P_2$

Epipolar Geometry
Figure 3-1

3.3 Background Theory

Given two stereo images I_1 and I_2 (the content of the focal planes P_1 and P_2 of the cameras), and two lines L_1 and L_2 contained in P_1 and P_2, in general every point of L_1 maps to a line segment in P_2, and there is no particular relationship between the line segments mapping to different points.

Fundamental property:

Every point in L_1 will map to a line segment contained in the same line L_2 if and only if L_1 and L_2 are a pair of conjugate epipolar lines.

Epipolar Geometry:

The family of epipolar lines in a focal plane is:

Either

a) a set of parallel lines having a common epipolar direction,

or

b) a bundle of lines, the intersection of which is the epipole.

3.4 Requirements of the System

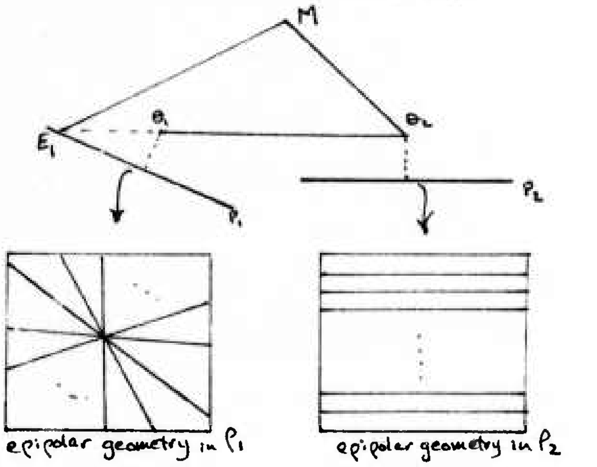
Given a pair of stereo images, we want to:

1) identify the kind of epipolar geometry present in the images;

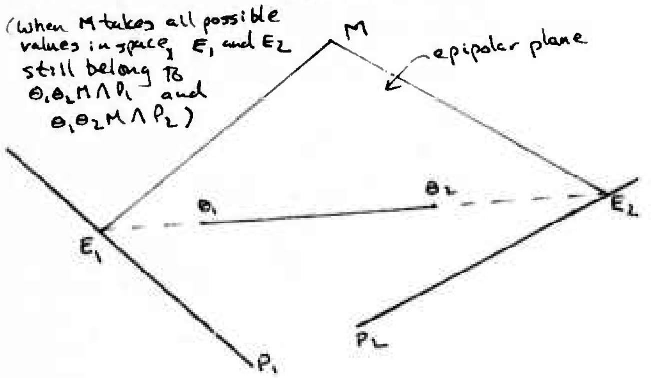
2) explicitly show the epipolar lines belonging to each image;

3) for each image, compute the parameters which relate the original coordinates to the epipolar coordinates;

4) construct the image transforms in epipolar space.



Top view of a situation with only one epipole
Figure 3-2


(when M takes all possible values in space, E1 and E2 still belong to θ1θ2M ∧ P1 and θ1θ2M ∧ P2)
M
epipolar plane
θ1
θ2
E1
E2
P1
P2

Top view of a situation with epipoles
Figure 3-3

Prior to these 4 steps, we will need to solve for the cameras, that is, to determine the 5 parameters describing their relative orientation. A procedure developed at Stanford [Gennery 1980] is used for this.

3.5 Algorithm Used

3.5.1 – Camera registration output

Each camera is viewed as a referential $(0_i, x, y, z)$, $i \in \{1, 2\}$. The registration procedure yields *azimuth, elevation, pan, tilt,* and *roll* of one camera with respect to the other. From there, we compute:

1) the rotation matrix R between the two referentials:

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

2) the translation unit vector t, the components of which are:

$$\begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} \text{in the base } 0_1xyz, \qquad \begin{pmatrix} \lambda \\ \mu \\ \nu \end{pmatrix} \text{in the base } 0_2xyz$$

Note that the magnitude of the translation vector cannot be determined from a pair of images.

3.5.2 – Epipolar geometry determination

The focal planes P_i are planes parallel to 0_ixy, intersecting 0_iz at $z = f_i$, the focal distance. There is an epipole in plane P_i if and only if the translation vector intersects this plane, that is, if and only if its third component is not zero.

We thus determine the case we are working with:

if $\gamma \neq 0$ and $\nu \neq 0$, there are two epipoles:	CASE 1
if $\gamma \neq 0$ and $\nu = 0$, there is one epipole E_1:	CASE 2
if $\gamma = 0$ and $\nu \neq 0$, there is one epipole E_2:	CASE 3
if $\gamma = 0$ and $\nu = 0$, there are no epipoles:	CASE 4

Remarks:

a) $\gamma = 0$ is replaced in the code by $|\gamma| <$ threshold, where threshold is chosen as a function of the arithmetic precision of the machine: if we had infinite precision, then we could consider every case as being case 1. Here threshold $= .0001$ was found to be a good estimate.

b) Most image pairs will belong to the first case, with γ and ν of the order of .1. The epipoles exist, are outside the picture frame, and, for the images worked with to date, tend to be at a distance of about 10 times the picture dimension.

3.5.3 – Epipoles and epipolar directions

If the epipole E_i exists, it is the extremity of the vector collinear to the translation vector, with a third component equal to f_i. If it does not, then the translation vector is the epipolar direction. Hence:

Case 1:	$E_1(\frac{\alpha f_1}{\gamma}, \frac{\beta f_1}{\gamma})$	$E_2(\frac{\lambda f_1}{\nu}, \frac{\mu f_1}{\nu})$
Case 2:	$E_1(\frac{\alpha f_1}{\gamma}, \frac{\beta f_1}{\gamma})$	$V_2(\lambda, \mu)$
Case 3:	$V_1(\alpha, \beta)$	$E_2(\frac{\lambda f_2}{\nu}, \frac{\mu f_2}{\nu})$
Case 4:	$V_1(\alpha, \beta)$	$V_2(\lambda, \mu)$

3.6 Epipolar line calculation

3.6.1 – CASE 1: two epipoles

a) theory

Let $M_1(x_1, y_1, z_1)$ be a point in P_1. E_1M_1 defines an epipolar line in P_1, and the corresponding E_2M_2 defines the conjugate epipolar line in P_2. The plane $(E_1, 0_1, 0_2, E_2, M_1, M_2)$ contains the translation vector t and E_1M_1. Its normal is $E_1M_1 \times t$. The normal of P_2 is 0_2z. Hence the intersection of the two planes is given by the vector:

$$0_2z \times (E_1M_1 \times t) = (0_2z \cdot t)E_1M_1 - (0_2z \cdot E_1M_1)t$$

and E_2M_2 is collinear to this vector. Suppose that in 0_1xyz, E_1M_1: $(x_1, y_1, 0)$. In terms of components in 0_2xyz:

$$0_2z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad t = \begin{pmatrix} \lambda \\ \mu \\ \nu \end{pmatrix}, \quad E_1M_1 = \begin{pmatrix} x'_1 \\ y'_1 \\ z'_1 \end{pmatrix} = \begin{pmatrix} r_{11}x_1 + r_{12}y_1 \\ r_{21}x_1 + r_{22}y_1 \\ r_{31}x_1 + r_{32}y_1 \end{pmatrix}$$

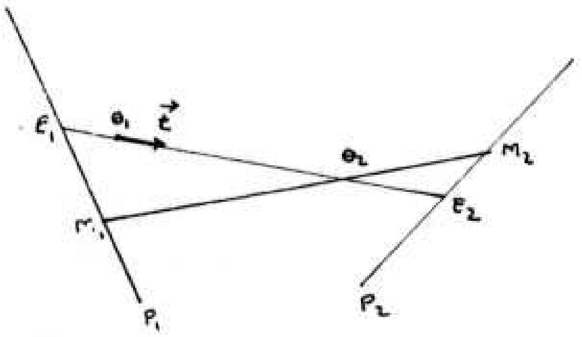
$E_2M_2$is collinear to:

$$\begin{pmatrix} \nu x'_1 - \lambda z'_1 \\ \nu y'_1 - \mu z'_1 \end{pmatrix} = \begin{pmatrix} x_1(\nu r_{11} - \lambda r_{31}) + y_1(\nu r_{12} - \lambda r_{32}) \\ x_1(\nu r_{21} - \mu r_{31}) + y_1(\nu r_{22} - \mu r_{32}) \end{pmatrix}$$

If we let Λ be the matrix:

$$\begin{pmatrix} \nu r_{11} - \lambda r_{31} & \nu r_{12} - \lambda r_{32} \\ \nu r_{21} - \mu r_{31} & \nu r_{22} - \mu r_{32} \end{pmatrix}$$

Then we can write $E_2M_2 = AE_1M_1$ where E_1M_1 is in base 0_1xy and E_2M_2 is in base 0_2xy.



Case 1
Figure 3-4

b) algorithm

Let N_1 be the number of epipolar lines that we want to determine. Each epipolar line is uniquely determined by the angle it makes with the x-axis. Let 0 be this angle. Given k, the epipolar line number, $0 \leq k \leq nl - 1$, how can we determine 0? If 0_0 and 0_1 are the lower and upper limits between which 0 is allowed to vary, and $0_2 = \frac{(\theta_1 - \theta_0)}{N_1}$, then we will choose the middle of each interval: $0 = 0_0 + (k + .5)0_2$ But what are 0_0 and 0_1? We have to distinguish between three cases:

- The epipole is in the picture (very unlikely). Then 0 can vary between 0 and 2π radians;
- The epipole is outside the image: there is a minimum and maximum angle under which the image is seen from this point. If we choose these angles in $[0, 2\pi]$ then most of the time every $0 \in [0_0, 0_1]$ will define a valid epipolar line in P_1;
- The exception from above is when the epipole is left of the image hut on a same vertical level: then $[0_0, 0_1]$ cannot be connected and still included in $[0, 2\pi]$. In this case we will choose the angles in $[\frac{-\pi}{2}, \frac{\pi}{2}]$.

Then L_1 will be defined by the point E_1 and the vector $V_1(cos0, sin0)$, and L_2 is defined by E_2 and $V_2 = AV_1$.

3.6.2 – CASE 2: one epipole E_1

a) theory

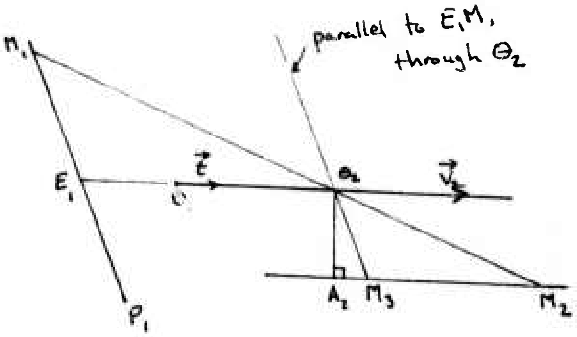
Given an epipolar line $\frac{L_1}{E_1M_1}$, we already know that the corresponding epipolar line L_2 in P_2 is collinear to the vector $t = V_2(\lambda, \mu)$. Hence we just need to find a point belonging to L_2. Clearly, L_2 is the intersection of P_2 with the plane $(E_1, M_1, 0_2)$. Thus any line contained in this plane will intersect P_2 at a point contained in L_2. In particular, consider the parallel to L_1 driven through 0_2. It intersects P_2, thus L_2, at M_3 such that, in base 0_2xyz:

$$E_1M_1 = \begin{pmatrix} r_{11}x_1 + r_{12}y_1 \\ r_{21}x_1 + r_{22}y_1 \\ r_{31}x_1 + r_{32}y_1 \end{pmatrix}, \quad \text{hence} \quad 0_2M_3 = \begin{pmatrix} f_2\frac{(r_{11}x_1 + r_{12}y_1}{r_{31}x_1 + r_{12}y_1)} \\ f_2\frac{(r_{21}x_1 + r_{22}y_1}{r_{31}x_1 + r_{32}y_1)} \\ f_2 \end{pmatrix}$$

b) algorithm

In the same way as in case 1, we define $L_1(E_1, V_1)$ where $V_1(x_1 = cos0_1, x_2 = sin0_2)$. Then L_2 is defined by (M_3, V_2), where the coordinates of M_3 are

$$\left(f_2\frac{r_{11}x_1 + r_{12}y_1}{r_{31}x_1 + r_{32}y_1}, f_2\frac{r_{21}x_1 + r_{22}y_1}{r_{31}x_1 + r_{32}y_1}\right)$$



Case 2
Figure 3-5

3.6.3 – CASE 3: one epipole E_2

a) theory

Let an epipolar line in P_1 be defined by the translation vector $t = V_1$ and by a point M_1 we pick. In P_2, L_2 goes through E_2 and is collinear to a vector E_2M_2, intersection of P_2 with the plane $(0_2, E_2, M_1)$. This plane is orthogonal to $0_2M_1 \times t$ and P_2 is orthogonal to 0_2z. Hence the intersection is collinear to $0_2z \times (0_2M_1 \times t)$

$$\begin{aligned} \text{since } 0_2M_1 &= 0_20_1 + 0_1M_1 \\ &= kt + 0_1M_1, \\ 0_2M_1 \times t &= 0_1M_1 \times t, \\ \text{and } 0_2z \times (0_2M_1 \times t) &= (0_2z \cdot t)0_1M_1 - (0_2z \cdot 0_1M_1)t. \end{aligned}$$

Suppose $0_1M_1 : (x_1, y_1, f_1)$ in 0_1xyz. Then in 0_2xyz:

$$0_2z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad t = \begin{pmatrix} \lambda \\ \mu \\ \nu \end{pmatrix}, \quad 0_1M_1 = \begin{pmatrix} x'_1 \\ y'_1 \\ z'_1 \end{pmatrix} = \begin{pmatrix} r_{11}x_1 + r_{12}y_1 + r_{13}f_1 \\ r_{21}x_1 + r_{22}y_1 + r_{23}f_1 \\ r_{31}x_1 + r_{32}y_1 + r_{33}f_1 \end{pmatrix}$$

E_2M_2 is thus collinear to:

$$\begin{pmatrix} \nu x'_1 - \lambda z'_1 \\ \nu y'_1 - \mu z'_1 \end{pmatrix} = \begin{pmatrix} x_1(\nu r_{11} - \lambda r_{31}) + y_1(\nu r_{12} - \lambda r_{32}) + f_1(r_{13} - \lambda r_{33}) \\ x_1(\nu r_{21} - \mu r_{31}) + y_1(\nu r_{22} - \mu r_{32}) + f_1(\nu r_{23} - \mu r_{33}) \end{pmatrix}$$

Hence, if we let A be the same matrix as in CASE 1 and OF3 be the offset matrix:

$$\begin{pmatrix} \nu r_{13} - \lambda r_{33} \\ \nu r_{23} - \mu r_{33} \end{pmatrix}$$

Then we have: $E_2M_2 = A0_1M_1 + OF3$

Where 0_1M_1 is in base 0_1xy and E_2M_2 is in base 0_2xy

b) algorithm

Now, how do we pick M_1 in the first place? We want a set of N_1 equally spaced epipolar lines, and it appears convenient to pick points on the axes. If the epipolar lines are more horizontal, or the image stretched in height, then we will pick N_1 equally spaced points on the vertical axis, suitably located to cover the entire image. If the epipolar lines are more vertical or the image more stretched in width, then we pick them on the horizontal axis. Let L_x, L_y be the picture dimensions and (V_x, V_y) the epipolar direction:

If $V_xL_y < V_yL_x$, and $V_y < 0$, we pick

$$y_1 = (L_x|\frac{V_y}{V_x}| + L_y)(\frac{K+0.5}{N_1})$$

If $V_xL_y < V_yL_x$, and $V_y > 0$, we pick

$$y_1 = (L_x|\frac{V_y}{V_x}| + L_y)(\frac{K+0.5}{N_1}) - L_x\frac{V_y}{V_x}$$

If $V_yL_x < V_xL_y$, and $V_y < 0$, we pick

$$x_1 = (L_y|\frac{V_x}{V_y}| + L_x)(\frac{K+0.5}{N_1})$$

If $V_yL_x < V^xL_y$, and $V_y > 0$, we pick

$$x_1 = (L_y|\frac{V_x}{V_y}| + L_x)(\frac{K+0.5}{N_1}) - L_y\frac{V_x}{V_y}$$

Then we proceed as indicated above:

$L_1(M_1, V_1 = t)$ is matched with $L_2(E_2, V_2 = E_2M_2)$.

3.6.4 – CASE 4: no epipoles

a) theory

Let an epipolar line in P_1 be defined by the translation vector $t = V_1$ and by a point M_1 we pick. In P_2, L_2 goes through the image of M_1, that is the extremity of a vector collinear to 0_1M_1 and whose third component is f_2. In base $0_1xyz, 0_1M_1 : (x_1, y_1, f_1)$. In base 0_2xyz:

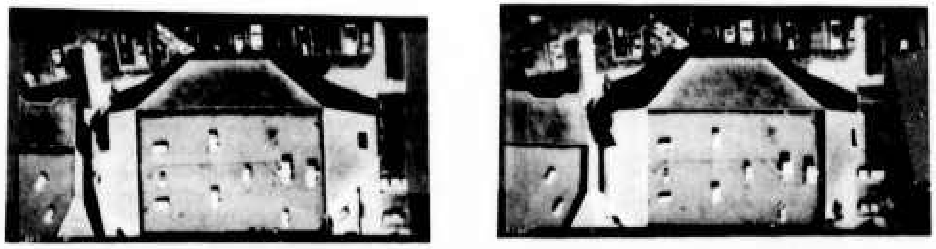
$$0_1M_1 = \begin{pmatrix} r_{11}x_1 + r_{12}y_1 + r_{13}f_1 \\ r_{21}x_1 + r_{22}y_1 + r_{23}f_1 \\ r_{31}x_1 + r_{32}y_1 + r_{33}f_1 \end{pmatrix} \quad 0_2M_2 = \begin{pmatrix} f_2\frac{r_{11}x_1 + r_{12}y_1 + r_{13}f_1}{r_{31}x_1 + r_{32}y_1 + r_{33}f_1} \\ f_2\frac{r_{21}x_1 + r_{22}y_1 + r_{23}f_1}{r_{31}x_1 + r_{32}y_1 + r_{33}f_1} \\ f_2 \end{pmatrix}$$

b) algorithm

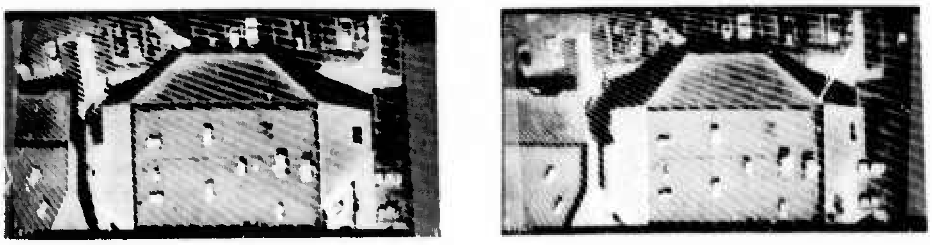
We pick points E_1 in the same manner as in case 3. Then we calculate 0_2M_2 as indicated above and we match $L_1(E_1, t)$ with $L_2(E_2, t)$.

3.7 Epipolar Registration and Transformation

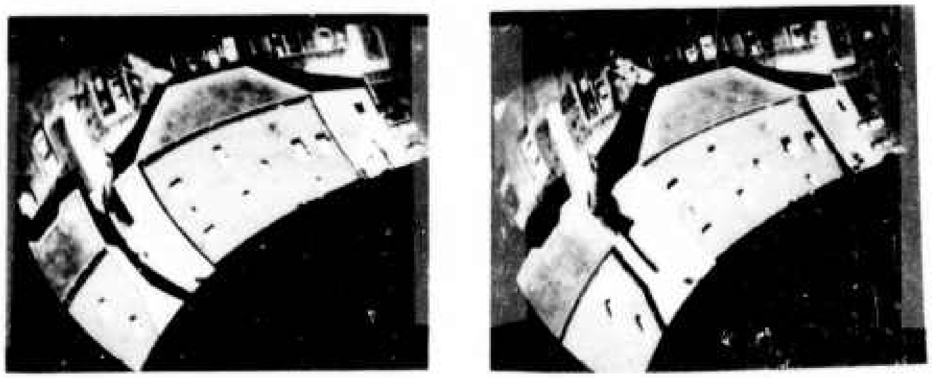
Figure 3-6 shows a stereo pair of a building complex. Figure 3-7 has this pair superpositioned with a set of corresponding epipolar lines. Figure 3-8 shows the imagery transformed such that epipolar lines are horizontal in the image, and conjugate epipolar lines have the same row coordinate.



Sacramento Building Image Pair
Figure 3-6



Epipolar Lines in this Imagery
Figure 3-7



Transformed Imagery
Figure 3-8

4: Automated Stereo Mapping

4.1 Background

Results from our laboratory over the past few years [Quam 1971, Hannah 1974, Moravec 1980, Gennery 1980, Arnold 1980, Baker 1981, Arnold 1983], have demonstrated the possibilities of both area-based and feature-based stereo matching.

Area-based stereo matching uses windowing mechanisms to isolate parts of two images for cross-correlation. *Feature-based* stereo matching uses two-dimensional convolution operators (and perhaps grouping operators) to reduce an image to a depiction of its intensity boundaries, which can then be put into correspondence. Area-based cross-correlation techniques require distinctive texture within the area of correlation for successful operation. In general, it breaks track where there is no local correlation (zero signal, or where two images do not correspond, *i.e.* occlusions) or where the correlation is ambiguous (where the signal is repetitive).

Demands of mapping in cultural sites and in locales with surface discontinuity and ambiguous or non-existent texture make it essential that, if area-based analysis is to be done, it be done in conjunction with feature-based analysis. *Feature*-based analysis provides a solution to many of the problems of correlation. Principal among its advantages is that it operates on the most discriminable parts of an image: places that are distinctive in their intensity variation, and where localization is greatest. These are typically the boundaries between objects or between details on objects, or between objects and their backgrounds. The important point is that the features being put into correspondence for depth estimates are the boundaries of objects: area-based analysis is at its worst at object boundaries, yet determining boundaries can be said to be the most important part of mapping in 3-space.

The [Baker 1981] system is the only current system that mixes these two matching modalities. We have been working at applying this system to some cultural scenes. Before carrying out these analyses we wished to:

a) enhance the system with a capability to work with a better edge operator [Marimont 1982];

b) enable it to process images that are not graced with collinear epipolar geometry (*i.e.* most images);

c) introduce an additional correspondence measure – edge extent.

4.2 Epipolar Registration

To implement these enhancements required substantial redesign of the system, and redesign cycles with the Marimont process. Closing useable data also presented difficulties, as the only imagery available was not of the correct geometry (see below). The two image pairs initially chosen (the Sacramento apartment complex and a section of some imagery of Moffett Field) proved, on closer examination, to require quite complex transformation, and could not be easily adjusted for epipolar processing.

In general, to bring imagery data into a properly transformed state could proceed in one of two ways:

- one could determine the transforms and then modify the imagery, producing an image pair having collinear epipolar geometry;

  or

- one could determine the transforms, and modify the output of an edge operator process that functions over the original imagery.

The latter is by far the superior approach, as it avoids resampling the image. This approach necessitates incorporating the transform computation into the stereo system, to follow edge finding and precede edge matching.

The second part of the stereo system's analysis is an intensity correlation process. This operates along epipolar lines as well, and clearly requires intensity information to be accessible along epipolar lines. One solution to this would be to take the original image pair and have the correlator rotate and change shape, size, and orientation as it moves around the image; this is an awkward and probably unnecessary complication. An alternative would be to access the transformed images, sampled as accurately as possible, and do the correlation in the rectangular space defined by collinear epipolar lines. The argument from edge accuracy indicated that transforming edges rather then resampling the image was the way to go; this argument from intensity correlation suggests that the resampled image can be useful.

Another implementation detail supported this use of both transformed edges and transformed imagery: it was found that the intensity information available from the Marimont process had too small a basis for useful correlation, and in fact, for transformed edges, had little relevance for the matching (it being measured not along epipolar lines, but normal to the edge direction). The transformed image had to be referenced again by the system to obtain more significant intensity estimates oriented along epipolar lines, and working with the image in epipolar space facilitated this.

The philosophy of the stereo matching process here had been to use edge analysis for, among other things, its higher accuracy, and to use intensity analysis for the continuity it provides. To be consistent with this, we wanted to have the highest possible accuracy for edges in epipolar space, and if sacrifice be needed for simplicity, to do it where it least degraded the analysis - in the intensity correlation. It is clear that transformed edges give higher accuracy than edges from transformed images (detectability might not change much, but localization is significantly reduced); and important simplifications could be obtained for little loss by doing the intensity correlation over the resampled image pair. This meant changes in our plans for the registration system: it had to produce not just transform information, but transformed images as well. Both forms are made available as output from the registration program described in section 3, and the enhanced Baker system uses them both.

4.3 The Marimont Edge Operator

The Marimont edge operator has greater detection and reliability than the original Baker edge operator, and similar localization; earlier examples of its processing convinced us that its output would improve the quality of our stereo reconstruction. Its ability to track along zero signal areas in following zero-crossing edges leads to more coherent image descriptions. [Marimont 1982] provides details of the operator's functioning. Roughly, it works by convolving an $m \times m$ lateral inhibition function of $n \times n$ central window with an image. Zero crossings in this resultant image then indicate edges, and the edge position is determined by interpolating over the lateral inhibition surface.

A few unanticipated problems became apparent once work with the edges was begun. One point, noted above, was that the intensity information stored at an edge (its left and right boundary values) had quite small support (a single pixel). This is in contrast with the original operator which interpolated for these values in an area 3 pixels wide and removed one pixel from the determined edge position. Another problem was that the edge connectivity produced by the Marimont system can be misleading. Intensity significance was improved by sampling along epipolar lines in the transformed images. The connectivity problem has not been looked at yet. Good connectivity is inherently difficult to achieve with zero crossing operators. Refinements to the process are being considered.

4.4 Edge Extent

The introduction of edge extent as a parameter in the dynamic programming solution was an obvious fallout from using the Marimont edges. Edges are output by that process as strings, with 2-connectedness. The maximum and minimum of some string, in transform space, is a measure of its (epipolar) extent. Prior to the use of this information the only way that global continuity entered the analysis was through a consistency enforcement relaxation process which ensured that edges connected in one view were interpreted as continuous in 3-space; all matching measures were quite local. With the modified approach, the correspondence measure is a function of (among other, more statistically based parameters) the ratio of edge extents. In particular, the likelihood of edge element a in the left image matching edge element b in the right image depends on the product of the ratios of the two upper extents (up from the edge elements) and the two lower extents (down from the two edge elements).

4.5 Testing on Imagery

When image testing began with all of the above accomplished, another problem became apparent: the stereo system, bound into a machine architecture with a maximum of 256K words of memory, and always tightly wedged anyway, had grown with these changes to the point that only small portions of images could be worked on at once. Thus came to exist a windowing mechanism within the edge finding/loading and stereo matching processes.

Our testing has been progressing on several sets of imagery: a synthetic image pair from Control Data Corporation, an aerial scene from the Engineering Topographic Laboratory, and a building scene of Sacramento. We will report on the results of these analyses at a later date.

5: Summary

A principal research interest of our group is in developing a rule-based advanced automated stereo mapping system to function within ACRONYM [Brooks 1981]. Current mapping techniques ignore much of the information available from inference on single views of a scene. This information can be useful for three-dimensional surface interpretation, and also provides extra parameters for stereo matching (*i.e.* surface orientation, occlusion cues). Our research effort is directed at establishing such monocular inference rules in a rule-base for stereo mapping.

In deriving these rules, we perform analysis of both hand extracted and automatically produced edge descriptions. A facility has been developed for this manual edge extraction from hardcopy imagery. We have studied rule synthesis for several cases, including that of orthogonal trihedral vertices - features that dominate cultural scenes. This research is very promising, and has shown the utility of the rule-based approach to surface inference from monocular information.

Camera solving provides powerful constraint on the correspondence problem in stereo matching. We have developed a facility for interactively registering images, determining the parameters for transforming them (or their edge descriptions) into collinear epipolar space, and performing the actual image transformation. This determination is crucial to a mapping process. Incorporating an automated module to provide data for the camera solving is a very important next step.

We have experimented with an existing stereo mapping process, enhancing its flexibility with respect to image format and with respect to edge operator format, and have been preparing example outputs of its processing on new imagery. Our intent with this effort has been to show the capabilities of a local matching process and to assess its applicability to the planned rule-based system.

Acknowledgements

This work was supported by Syracuse University Subcontract PO#29205485 (Air Force F30602-81-C-0169), NASA NAG 5261, and ARPA Contract N0039-82-C-0250.

6: References

[Arnold 1980] Arnold, R.D., and T.O. Binford, "Geometric Constraints in Stereo Vision," *Soc. Photo-Optical Instr. Engineers,* **Vol.** 238, Image Processing for Missile Guidance, 1980, 281-292.

[Arnold 1983] Arnold, R. David, "Automated Stereo Perception," Department of Computer Science, Stanford University, Ph.D. thesis, 1983.

[Baker 1981] Baker, H. Harlyn, "Depth from Edge and Intensity Based Stereo," University of Illinois, Ph.D. thesis, September 1981.

[Binford 1981] Binford, Thomas O., "Inferring Surfaces from Images," *Artificial Intelligence,* **Vol.** 17(1981), August 1981, 205-244.

[Brooks 1981] Brooks, Rodney A., ' Symbolic Reasoning Among 3-D Models and 2-D Images," Stanford Artificial Intelligence Laboratory, AIM 343, June 1981.

[Duda 1973] Duda, R.O. and P.E. Hart, **Pattern Classification and Scene Analysis**, Wiley, New York, 1973.

[Gennery 1980] Gennery, Donald B., "Modelling the Environment of an Exploring Vehicle by Means of Stereo Vision," Ph.D. thesis, Stanford Artificial Intelligence Laboratory, AIM 339, June 1980.

[Hallert 1960] Hallert, Bertil, *"Photogrammetry, Basic Principles and General Survey,"* McGraw-Hill Book Company Inc., 1960.

[Hannah 1974] Hannah, Marsha Jo, "Computer Matching of Areas in Stereo Images," Ph.D. thesis, Stanford Artificial Intelligence Laboratory, AIM-239, July 1974.

[Liebes 1981] Liebes Jr., S., "Geometric Constraints for Interpreting Images of Common Structural Elements: Orthogonal Trihedral Vertices," *Proceedings of the DARPA Image Understanding Workshop,* 1981.

[Lowe 1982] Lowe, David G., "Segmentation and Aggregation," *Proceedings of the DARPA Image Understanding Workshop,* Stanford University, September 1982.

[Marimont 1982] Marimont, David H., "Segmentation in ACRONYM," *Proceedings of the DARPA Image Understanding Workshop,* Stanford University, September 1982.

[Moravec 1980] Moravec, Hans P., "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," Stanford Artificial Intelligence Laboratory, AIM-340, Ph.D. thesis, September 1980.

[Perkins 1968] Perkins, "Cubic Corners," in Quarterly Progress Report, MIT Electronics Laboratory, April 15,1968

[Quam 1971] Quam, Lynn H., "Computer Comparison of Pictures," *Stanford Artificial Intelligence Laboratory, AIM-144,* Ph.D. thesis, 1971.

A CONTENT ADDRESSABLE ARRAY PARALLEL PROCESSOR AND SOME APPLICATIONS

Charles Weems
Steven Levitan
Daryl Lawton
Caxton Foster

Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

ABSTRACT

A design is presented for a Content Addressable Array Parallel Processor (CAAPP) which is both practical and feasible. Its practicality stems from an extensive program of research into real applications of content addressability and parallelism. The feasibility of the design stems from development under a set of conservative engineering constraints tied to limitations of VLSI technology. We then describe the implementation of two procedures for image processing on the CAAPP. The first performs image convolutions very quickly. It is shown that this algorithm can be generalized to perform convolutions with increased mask size with only a moderate reduction in speed. The second uses the CAAPP to quickly and robustly decompose an optic flow field into its rotational and translational components to recover sensor motion parameters. It is important to note that this latter is made possible only by the combination of associativity and array processing that our design provides.

1.0 DESIGN OF THE CAAPP

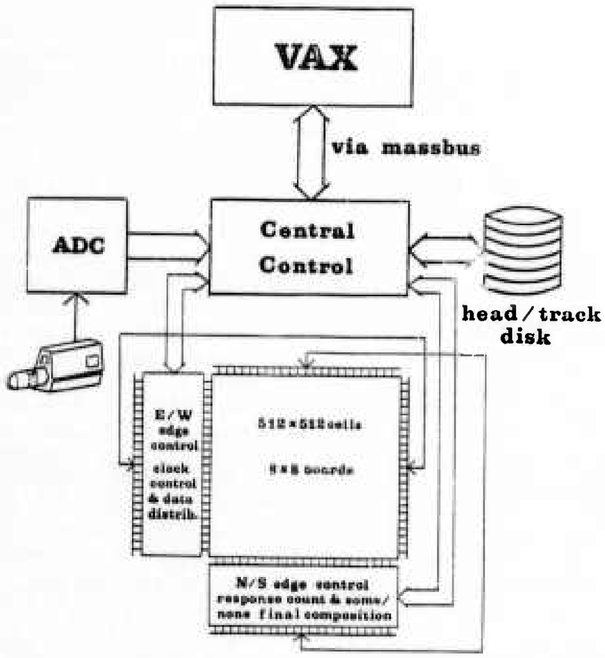
We have developed a design for a VLSI-based Content Addressable Array Parallel Processor (or CAAPP) for image processing and other applications. Our intention has been to produce a feasible design which would be simple enough for us to construct with reasonable confidence of success but which would also provide a significant advance in processing power. Accordingly, a number of constraints have been imposed on the design. The CAPP would have to consist of no more than one hundred circuit boards and each board should have a maximum of one hundred off-board connections. Additionally, the VLSI chips we were to design would be restricted to existing feature and die sizes, have a pin-out of no more than forty pins, and a power dissipation of less that two watts.

We also set a number of goals which we hoped to achieve. It was decided that the CAPP should contain 262,144 cells arranged as a rectangular 512x512 array to facilitate image processing. Each cell would contain at least thirty-two bits of memory (preferably sixty-four bits), multiple tags, and some bit serial processing power. One hundred nanoseconds was set as a goal for the minor clock cycle time. Additionally, for image processing applications, we needed to be able to load the memory with an image in one video frametime (1/30 second). For sixteen-bit pixels this means a data transfer rate of about sixteen million bytes per second. Finally, it was decided that the CAPP would be built to be driven by another machine, such as a Digital Equipment Corporation VAX. Once the goals and constraints were set, work on the design got under way.

1.1 The CAAPP and Its Environment

The CAAPP is divided into two main parts: the central control and the parallel processor (See figure 1). The central control is a simple, fast, fetch-ahead pipelined processor which will be built from MSI devices. It issues instructions to the parallel processor, controls loading and unloading of data in the parallel processor, serves as an interface to the VAX or other host computer and to other data sources and secondary storage devices. The central controller contains a ROM with a set of micro-coded subroutines for performing commonly needed higher level CAAPP operations, and a writeable control store which allows users to add their own special microcoded instructions. Also contained in the central controller is a small program memory into which subroutines or entire programs may be loaded. The writeable control store and program memory are loaded directly by the VAX. Once these memories are loaded, the VAX can issue commands to the central controller which tell it to execute routines stored in the program memory, to single step through a stored routine, or to execute a single instruction passed as a literal by the VAX.

This work was supported in part by grants from the Army Research Office DAAG29-79-G-0046 and grant number N00014-82-K-0464 from DARPA.



VAX
via massbus
ADC
Central Control
head / track disk
E/W edge control
clock control & data distrib.
512×512 cells
8×8 boards
N/S edge control response count & some/none final composition

Figure 1

1.1.1 The Parallel Processor

The parallel processor design consists of an 8x8 array of processing circuit boards and a set of special purpose boards which control how the edges of the CAAPP are treated. The parallel processor receives data and instructions broadcast to it by the central controller. Each parallel processor instruction operates in exactly one minor cycle time. Some operations do require multiple clock cycles, but these are taken care of by having the central control rebroadcast the instruction as many times as necessary.

Each processor board consists of an 8x8 array of special CAAPP integrated circuits plus some random buffer logic. Our current design calls for all sixty-four processor circuit boards to be placed in four card racks (sixteen per rack) and interconnected by a broadcast backplane and ribbon cables.

The heart of the CAAPP design is a special purpose nMOS VLSI integrated circuit. Each of these chips will contain sixty-four CAAPP cells, an instruction decoder, and other miscellaneous logic. We have designed the chip with as much generality as possible, knowing that such generality need not be fully used later on.

1.1.2 The Communications Interconnect

One of the biggest problems in designing the CAAPP was how to handle the rectangular interconnection of the cells. The number of wires required for such a network, even for bit serial communications, is staggering. This became most evident when we tried to design the IC communications interface. For sixty-four cells, the arrangement which gives the minimum number of external connections is an 8x8 grid. With a four-way N,S,E,W interconnect there are then only thirty-two neighboring cells to connect to. (We considered an eight-way N,S,E,W,NW,NE,SW,SE interconnect, but were forced to abandon it due to the wiring complexity.) By the time control, power, and clock signals were added to the thirty-two neighbor lines, we found that a sixty-four pin package would be required to hold the IC. Further examination also revealed that full interconnection would require that each processor board have 256 ribbon cable communication lines -- in other words, a two foot wide swath of ribbon cable running between each pair of boards! Because this violated two of our main design constraints, we had to simplify the interconnections.

By 8:1 multiplexing the communications net as it crossed chip boundaries, we were able to reduce the IC pin count to twenty-two pins and the I/O lines per board to sixty-three (of which only thirty-two need to be run in ribbon cable, the rest being backplane bus signals). By going from sixty-four pin to twenty-two pin packages, the board size was also reduced significantly. Unfortunately, all of these benefits were paid for in a loss of speed. The new interconnect takes 0.8 microseconds to transfer one bit between cells (25.6 microseconds for thirty-two bits). We should also note here that the CAAPP instruction set makes this multiplexing transparent to the user.

1.1.3 Some/None Logic

A common means of controlling loop execution in CAM algorithms is to continue processing until none or only one of the CAM's tag bits are turned on. It is thus essential that we have a fast means of determining this. The simplest way of doing this is to test whether any tags are on; if so, we find the first one and turn it off, then repeat the some/none test. The "find-first" operation is also used frequently when a search selects several data elements with the same key value. Find-first then provides a way to select one of these for processing. Thus the need is great for fast some/none and find first operations. On-chip the Some/None signal is determined by feeding the output of the main tag bit into a sixty-four-way NOR with an inverter between its output and the Some/None pad driver. Once the signal goes off-chip, it passes through a four-level OR tree before reaching the central controller.

1.1.4 Count Responders

Many CAM designs devote much complex and expensive hardware to increasing the speed of the operation which counts the tag bits that are turned on. We have found, however, that the count of responding tags is used primarily as a way of gathering statistics for use at much higher levels of processing control to direct the strategic application of the CAM. It is thus rather infrequently applied as compared to operations such as comparisons and some/none tests. We thus feel that slower, simpler, less expensive response count hardware is quite acceptable. Further, we have designed a very simple response count system which runs only about an order of magnitude slower than much more complex designs.

The count responders operation requires only three changes to be made to the CAAPP circuitry to be feasible. First, it must be possible to connect all of the response bits on a chip into a circular shift register. (This is also useful as a means of testing the integrated circuits since it allows us to directly examine register contents with a test circuit.) This shift register is easily added because the neighbor communication network already provides most of the necessary links. Secondly, a register, a counter, and a full adder must be added to each chip. Finally, the cards that control the top-bottom edge treatment must be modified to include column summing registers and a final sum register. The algorithm used is reasonably fast (about twenty-six microseconds), inexpensive, and most importantly it can be used with any size of array without having to modify the basic IC -- only the bottom row circuit board needs to be changed.

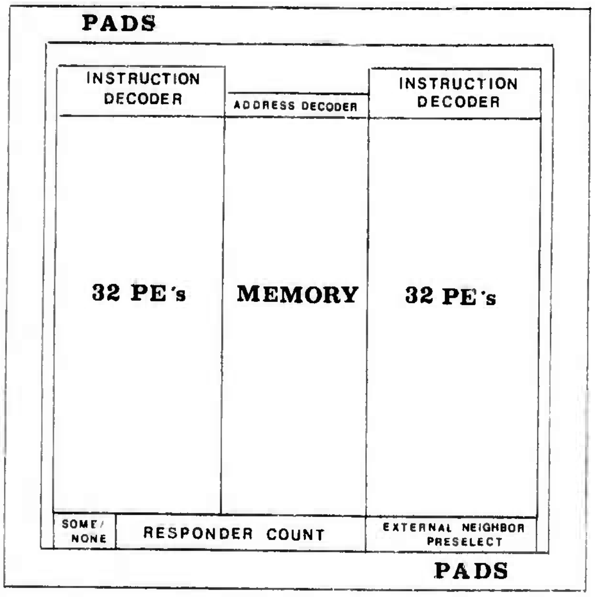


Figure 2

1.2 Device Floorplan

The unit cells are arranged in two columns of thirty-two (See figure 2). This arrangement was chosen because we found that the best compaction would be obtained if we could share control and memory select lines among as many cells as possible. Each cell is thus very long and narrow. A column of thirty-two cells is almost covered by a river of metal control and select lines which run vertically over it. These lines are simply duplicated and mirrored for the two columns. Control is generated by a NOR-NOR network forming the instruction and address decoders. Responder count hardware is provided in a small block of random logic. The overall size estimate of the active chip area (excluding pads and drivers) is 2400x2400 lambda. Thus if lambda is three microns, the central portion of the die would be roughly 285 mils on a side. This is somewhat large, but not unreasonable. Access to a fabrication facility with slightly smaller feature size and/or two layers of metal would significantly reduce the die size. Power dissipation is estimated at 1.5 watts, which is low enough to allow forced-air cooling.

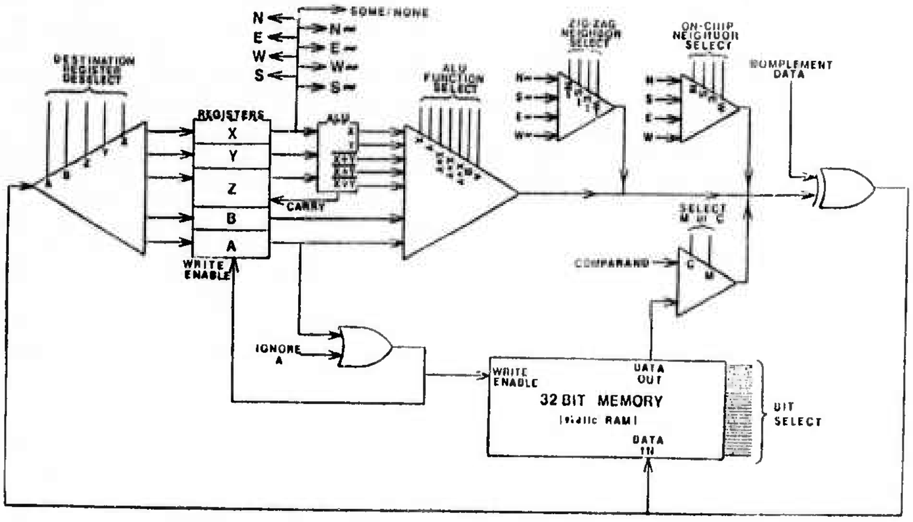
1.3 The Unit Cells

A unit cell consists of thirty-two bits (which will be expanded to at least 64 bits in the final design) of fully static memory, four one-bit static tag "registers" called A, B, X, and Y, and a static carry bit "register" called Z. Each cell also contains an ALU which continuously generates X nand Y, X nor Y, and X + Y + Z. Finally, each cell contains logic for selecting one source of data (a register, memory, an ALU function, broadcast data, or a neighbor cell), possibly inverting the selected signal and storing it in a destination (memory or register). Neighbor communication lines run vertically in two channels in the middle of the cell. The Z register is special in that it is not available for selection as a data source. It can be copied directly to the X register and can be loaded from the output of the selector. It also is loaded with the carry from X + Y + Z whenever that function is selected.

The X register is special in that its output is connected to the some/none logic and the neighbor communication network. In some sense it is the "main" tag bit.

The A register is also special. It controls whether the cell is active. If a cell is not active, it ignores all instructions broadcast by the central controller except a special few.

The Y register is intended to be used for storing a second set of tag bits which may eventually be combined with other sets through the logical operations provided by the ALU.



Organization of One PE

Figure 3

The B register is intended as temporary storage for a second set of activity bits, essentially providing a single level of "subroutine call" or an alternative activity "screen".

Figure 3 shows the logical arrangement of a unit cell.

1.4 CAAPP IC Instruction Set

Each CAAPP IC instruction executes in one minor clock cycle (100 ns). This was done to avoid feedback loops in the decoder on the chip and to avoid special instruction states in the central controller. This means that the central controller must be programmed to re-issue some instructions several times. For example, transferring data to neighbor cells across chip boundaries requires eight individual transfers because of the 8:1 multiplex. The central control must therefore issue the shift instruction eight times in a row.

There are eight basic instructions recognized by the chip. Of these, six are memory transfer operations and use a five-bit address value (this will be expanded to at least six with the increase in memory size) to select the bit to be read or written. The other two instructions treat the address as a sub-operation specifier. For the most part these are non-memory-data-source to register transfer operations with one op code causing the data to be inverted before storage and the other causing a direct transfer. There are nineteen special sub-ops, however, which are reserved for unusual operations such as transferring data on and off the chip or counting responders.

Some operations are also designated as "jam transfers". This means that they are performed regardless of whether the A register contains a logic one. These provide a means of storing and retrieving different activity patterns and of applying global operations which ignore activity without the usual overhead of having to save the current activity pattern, and retrieve it later.

1.5 Current Status

As of this writing we have designed a sixteen-cell (4x4) test chip. Using a simple set of three micron design rules, we have succeeded in fitting the circuitry onto a 180x180 mil body area with room to spare. The actual cell area occupies only 110x150 mils. Estimated power dissipation is only 350 milliwatts. The design includes about 7000 transistors. The memory portion of the chip is being fabricated as part of a student project by MOSIS. We are working closely the General Electric Cooporation on a feasability study examining the technologies they could bring to bear should we embark on a cooperative development project.

We have already written a number of programs for the CAAPP and estimated their operation times by hand. For example, one special purpose convolution of interest in computer vision processing (a simple 3x3 mask) required only 100 microseconds for the entire 512x512 image. (This will be presented in the next section.) More complex convolutions take longer, of course, but most of interest can be performed in less than five milliseconds. We have also examined motion analysis and found the results to be quite encouraging (some of which will be discussed later). An instruction level simulator has been programmed for the CAAPP and is currently being used to develop and test small applications, and to gather statistics on instruction set usage and predict execution times. (It is, of course, nearly impossible to test any really large applications on the simulator because most of these would require several days of computer time.) A higher level simulator is also being integrated into our department's computer vision research system so that researchers can begin to develop vision related algorithms for the CAAPP.

2.0 IMAGE CONVOLUTION ON THE CAAPP

Our previous work on Conway's Game of Life implemented on a CAM [2] demonstrated that such a device could be effectively used to speed up algorithms which dealt with rectangular grids of cells and small neighborhoods about each of those cells. Conway's Game of Life actually involves performing a very simple image convolution and the technique developed for it was applied to more general convolutions. This method was further refined with the CAAPP design.

2.1 Basic Technique

One simple implementation of convolution involves each cell on a rectangular grid examining its immediate neighborhood and then updating its own contents based upon some function of that neighborhood. The update must, of course, be performed after all cells have finished examining their neighborhoods. On a parallel array processor this examination can be performed simultaneously by all of the cells on the grid, as can the update operation. Thus the algorithm for the convolution can be described as the actions of a single cell with the understanding that each action is performed simultaneously by all of the cells.

There are two different ways of approaching the problem of examining the neighborhood. The one that first comes to mind is that each cell "looks" at each cell in its neighborhood, gathering what information it needs to perform an update. In practice this involves moving data from each cell in the neighborhood into the "central" cell where some function is then applied to it and the result stored for the update phase of the convolution. The problem with this is that the data must often pass through other cells before it reaches the central cell. For example, when the neighborhood is 7x7 cells, data from the outer ring of cells must pass through at least two other cells before reaching the center cell. Because movement of data takes time, this "passing through" is rather inefficient. The solution is to have the data stored in the intermediate cells on its way to the center, thus taking advantage of the fact that those cells will also need to know the values in order to compute the function of their neighborhoods. Although this will work, the algorithm becomes rather messy since we must now consider the actions of several cells at once and how these relate to each other. It also becomes a complex problem to determine an optimal set of data collection paths as the neighborhood's diameter varies.

It turns out that the other approach to examining the neighborhood greatly simplifies these problems. This approach takes the opposite view of the collection process. Instead of each cell collecting all of the data from its neighborhood, each cell distributes its own data to every cell in the neighborhood. Because every other cell is also doing this, the end result is that the central cell (and hence all cells) gets the data it needs from all of the cells in the neighborhood. The problem of establishing an optimal distribution path is trivial for a square array of odd diameter: It is simply a rectangular spiral out from the center cell. For even diameter square neighborhoods the problem is only slightly more difficult because the center cell is actually half of a cell width off center in two directions. In this case it is simply required that the appropriate choice of initial direction and of clockwise or counter clockwise spiral be made to select the optimal path. The only other point that requires mentioning is that, because this is a distribution process rather than a collection process, the function mask for the convolution must be mirrored across the central cell. For example, when the cell's value is being stored in its north neighbor, the function applied to that value is the south neighbor function. The reason for this can be seen when it is realized that the central cell is actually the south neighbor of the cell to its north. The mirroring of the convolution function mask is actually quite easy to accomplish: we simply step through the mask in exactly the opposite direction that the distribution path takes.

Let's look at an example: A simple convolution for smoothing isolated cells of noise out of an image. We will use a 3x3 convolution mask in which the cell accumulates the sum of its neighbor's values, weighted inversely with distance away from the center. The sum will then be normalized. Define the mask to be an array M

$$M = \begin{array}{c|ccc} & j & 0 \quad 1 \quad 2 \\ \hline i & & \\ 0 & 1 & 2 & 1 \\ 1 & 2 & 4 & 2 \\ 2 & 1 & 2 & 1 \end{array}$$

Where $M_{1,1}$ is the central cell. Then the following algorithm will perform the convolution (north is up, west is to the left, etc.):

```
i := 1
j := 1
sum := value *$M_{ij}$
move value north
i := i+1
sum := sum + value * $M_{ij}$
move value east
j := j+1
sum := sum + value * $M_{ij}$
move value south
i := i-1
sum := sum + value * $M_{ij}$
move value south
i := i-1
sum := sum + value * $M_{ij}$
move value west
j :+ j-1
sum := sum + value * $M_{ij}$
move value west
j := j-1
sum := sum - value * $M_{ij}$
move value north
i := i+1
sum := sum + value * $M_{ij}$
value := sum * normalizing factor
```

It should be noted that the time required to perform a convolution using the parallel processor is independent of the size of the image and only dependent upon the area of the convolution mask. Since the CAAPP does cell level arithmetic bit-serially, the size of the data values also affects the speed of the algorithm. It has been determined that convolution with a 3x3 mask requires 980 CAAPP operations. This corresponds to 98 micro-seconds per convolution or 340 convolutions per frame time or 10204 convolutions per second.

Convolutions with more general and/or larger masks will take longer. A very rough worst case estimate of the time required for such convolutions can be obtained from the formula:

$$T = P(.8N+.2M+.1) + .3M(N^2P+N+1)$$

where

- T = time in microseconds
- N = number of bits in a pixel
- M = number of bits in a mask value
- P = number of pixels in the mask area

This is a worst case time which assumes that all of the bits in all of the mask values are ones (since this gives the slowest multiply speed). Under normal circumstances, T will be about half of the value obtained from the formula. This also assumes a totally general square mask where the values can change. If constants are to be used for the mask values, a significant speed increase can be obtained by optimizing the multiplies for those values. Thus, for example, a convolution on 16 bit values with 8 bit mask values could be applied over at most a 7x7 mask in one video frame time with a worst case situation. For normal situations, it should be possible to convolve a 10x10 area. Given constant mask values, and depending upon the amount of optimization possible, even a 25x25 mask could be done in one video frame time.

As a final note, this method is not restricted to square masks and in fact should be readily generalizeable to any mask shape. All that is required for this is an algorithm for efficiently shifting the center cell's value so that it covers the mask area. Thus it should be possible to easily adapt it to such mask shapes as annuli and disjoint areas.

3.0 OPTIC FLOW FIELD DECOMPOSITION ON THE CAAPP

The realization of motion perception in artificial systems will require highly parallel architectures. We have explored the use of the CAAPP as an effective means of quickly and robustly decomposing a flow field into its rotational and translational components [4,5] to recover the parameters of sensor motion.

The algorithm is an exhaustive search procedure which uses a set of rotational and translational flow field templates to find a component pair which can account for the motion depicted in a given flow field. Currently, 1000 rotational templates and 200 translational templates are used. These are generated from 100 axes which are uniformly distributed with respect to a unit hemisphere, and all pass through the origin of the sensor coordinate system. Each flow field consists of 16x16 vectors and is stored on a 2x2 square of chips consisting of 256 cells. The 2x2 chip arrangement facilitates flow field addressing. Each cell contains the horizontal and vertical components of a flow vector, each specified with 10 bits of precision.

The algorithm consists of four basic steps.

(0) The rotational templates are loaded into the CAAPP, one template for each flow field location. Each flow field location corresponds to one of the squares in the CAAPP diagrams shown in Figures 5a, 5b, and 5c. The rotational templates need only be loaded once since they are used in determining any flow field decomposition.

(1) A copy of the input flow field is loaded into each flow field location in the CAAPP. Figure 4a and 4b show two sample input fields, both produced by the same motion and environment, except that Figure 4b was produced by adding random spike noise to Figure 4a.

(2) A set of difference fields is formed by subtracting each rotational template from the copy of the input flow field stored with it. For each resulting difference field, the slope of each difference vector is computed by dividing the vertical component by the horizontal component. These subtraction and division procedures are performed in parallel across all flow fields represented in the CAAPP.

(3) The similarity between the difference fields and each of the translational templates is evaluated, proceeding sequentially through all the translational templates. For a given translational template, this comparison is done in parallel with all difference fields stored in the CAAPP and consists of the following steps:

(3a) The slope of each component vector of the translational template is loaded into the corresponding vector location of each difference field. The sign of the slope of each difference vector is compared with the sign of the slope of the corresponding translational template vector. If the signs agree, the absolute value of the difference between the slope of the difference vector and the slope of the translational template vector is computed, and then scaled according to the absolute value of both slopes. If the scaled slope difference does not exceed a predetermined maximum error value, then a vector match is designated at that position. The quantity of error permitted here allows the algorithm to be resistant to uniformly distributed Gaussian noise of low variance present in the original flow field.

(3b) For each difference field the number of vector slope matches is counted. If this sum exceeds a predetermined minimum number of matches (in our implementation, 75% of the field size), then the associated rotational and translational templates become a candidate pair for the flow field decomposition. Utilization of a minimum number of required matches ensures that only templates which are reasonably close to the actual motion will be chosen and permits some resistance to random spike noise. Figure 5a shows, for difference fields resulting from the input field in Figure 4a, the CAAPP response to the translational template which is closest to the actual translational motion. Each black dot within a square represents a position in a difference field at which the slope of the difference vector matches the slope of the translational template. Figure 5b shows, for difference fields resulting from the input field in Figure 4b, the CAAPP response to the translational template which is closest to the actual translational motion. Figure 5c shows the CAAPP response to a translational template which is not close to the actual translational motion. This incorrect translational template is shown in Figure 6.

(3c) For all difference fields yielding at least the required minimum number of matches, the variance of the scaled slope difference is computed, and the difference field with the minimum variance is determined. This value is compared to the minimum variance found from processing the preceding translational templates. If this value is less than the preceding minimum, it becomes the new global minimum, and the rotational template associated with the difference field together with the current translational template become the current best candidate pair for the flow field decomposition.

Steps 3a, 3b, and 3c are performed for each translational template.

(4) The flow field decomposition considered to be the best is the rotational and translational template pair resulting in the difference field yielding at least the required minimum number of matches and the least slope difference variance. Utilizing minimum variance instead of the maximum number of matches, the algorithm has achieved better results, particularly for motions whose component parts lie between sets of templates. Figures 7a and 7b show the rotational and translational templates selected by the algorithm in the presence of and in the absence of noise, for the input fields in Figures 4a and 4b. These templates are the closest ones to the actual motions. Figures 8a and 8b show the difference fields resulting from subtracting the rotational motion in 7a from the original fields in Figures 4a and 4b respectively.

3.1 Flow Field Decomposition Experiments

Experiments have been performed with a CAAPP simulator on a VAX 11/780 using a wide variety of motions and simulated environments. In all cases examined, the translational template closest to the actual translational motion was selected. The rotational template was always close to the actual rotational motion, but was sometimes not the closest template. The procedure proved to be resistant to limited Gaussian noise as well as to limited random spike noise in the original flow field. Applying motion to points at random depths produced results similar to those obtained in the noise experiments. The algorithm's performance degraded slightly if each flow vector component was specified by eight bits of precision instead of by ten.

The CAAPP timing calculations revealed that the algorithm could perform the rotational-translational decomposition in slightly more than 1/4 second. If two CAAPPs are used in parallel, then the time can be reduced to less than 1/5 second, since only half of the translational templates need be tested on each CAAPP. Given fabrication techniques available in the immediate future, we expect execution times to be significantly improved. We suspect that performance will improve and be applicable to more realistic image sequences by increasing both the number and size of the rotational and translational templates. This amounts to utilizing more CAAPPs in parallel.

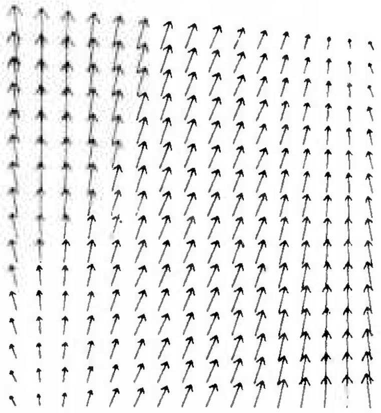


Figure 4a

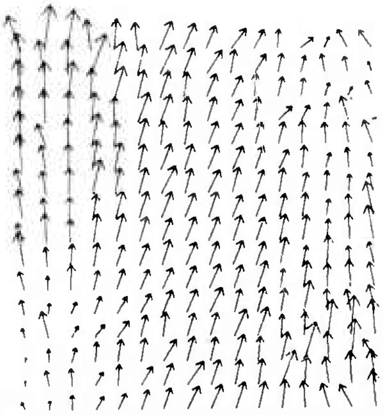


Figure 4b

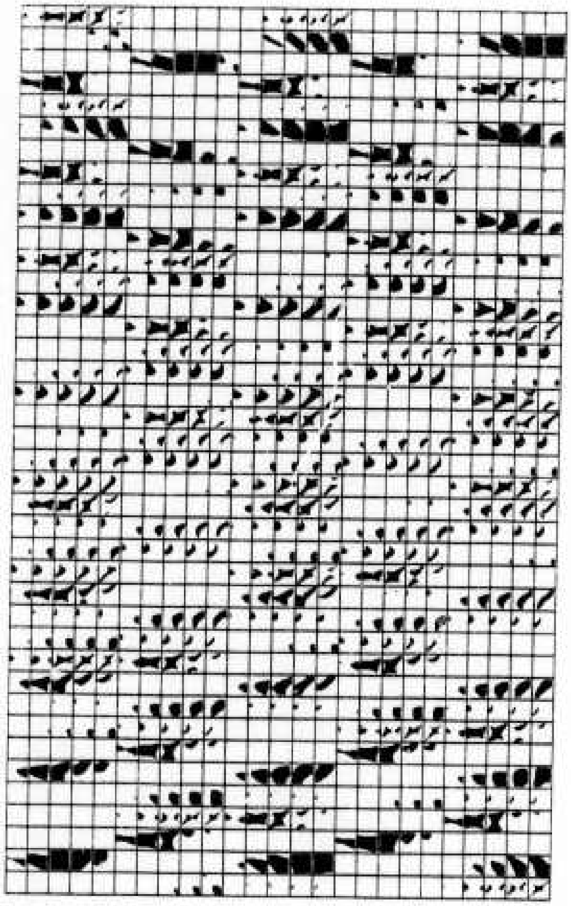


Figure 5a

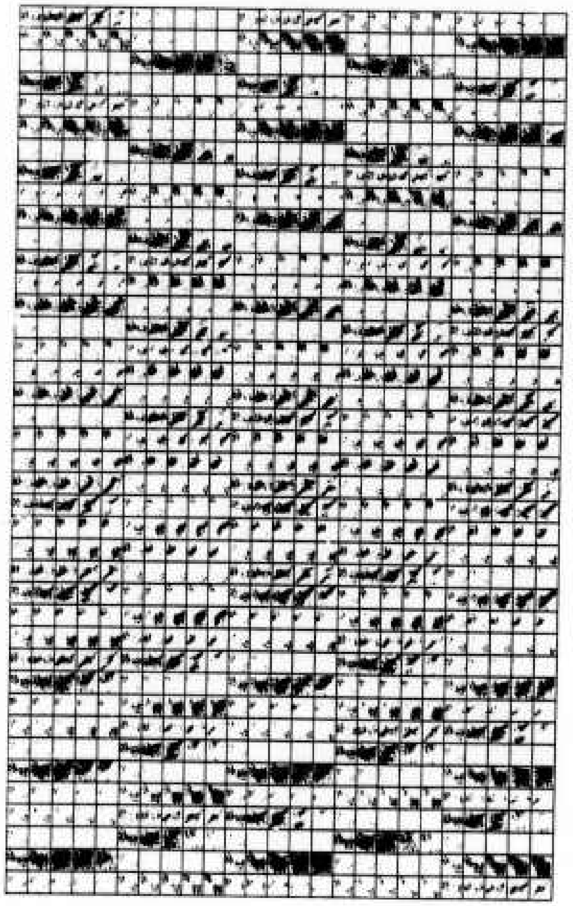


Figure 5b

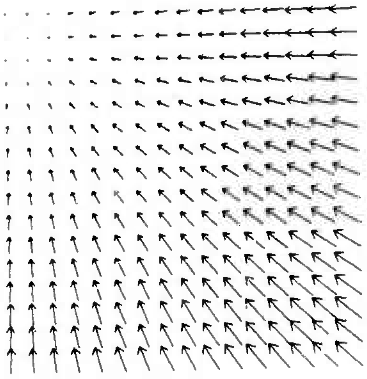


Figure 6

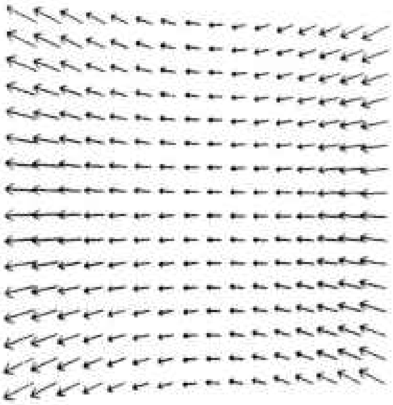


Figure 7a

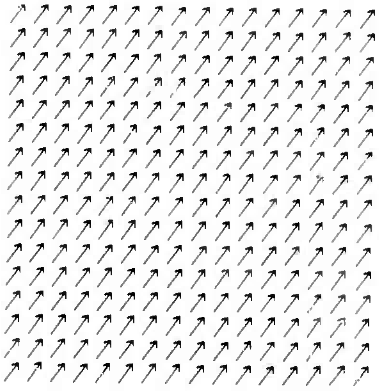


Figure 7b

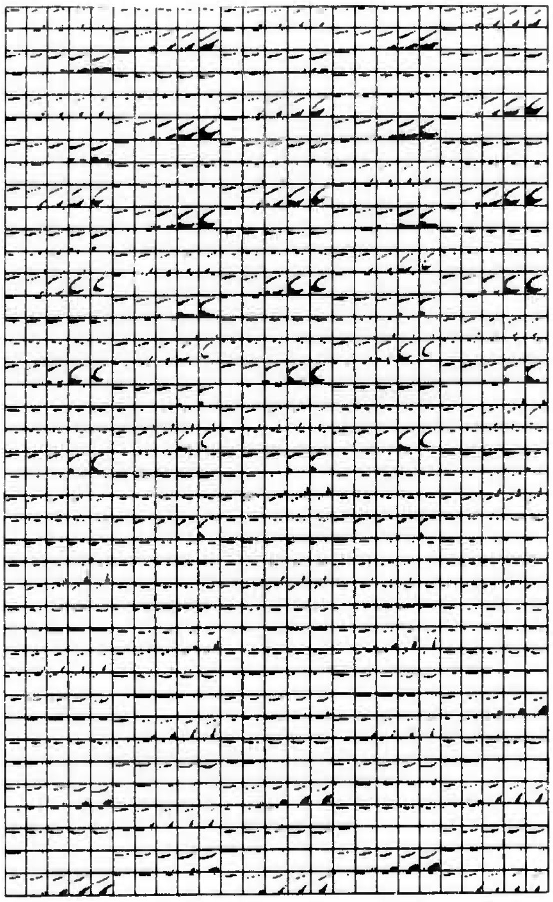


Figure 5c

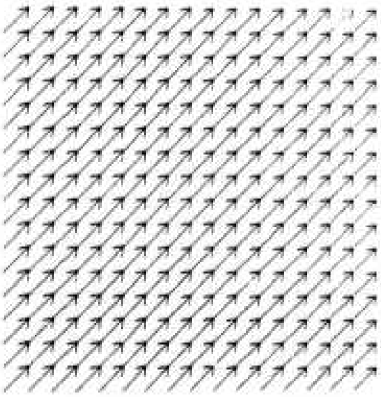


Figure 8a

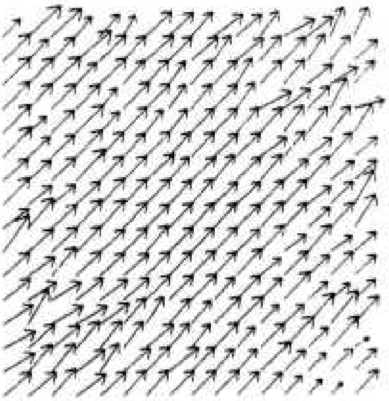


Figure 8b

4.0 CURRENT RESEARCH

Based on the results of our test chip experience, we intend to proceed to full sixty-four cell ICs and, eventually, construction of the entire machine. Architectural changes which we intend to pursue are increasing the memory size to at least sixty-four bits per cell and perhaps going to an 8:2 communications multiplex (with a twenty-eight pin package) for a doubling in the data transfer rate.

Our work thus far has indicated that a Content Addressable Array Parallel Processor is well suited for many aspects of image processing, vision, and motion analysis. We are exploring the effective implementation of a wide range of image processing algorithms on the CAAPP [6]. We intend to pursue further applications in these areas and also in new areas such as tactile object recognition in robotics.

5.0 CONCLUSIONS

A design has been presented for a Content Addressable Array Parallel Processor suitable for both general use and image processing applications. The architecture of the processor is based in practical experience and the hardware design has been constrained to make it possible to construct using existing technology and with a high confidence of success. Despite these constraints, simulations have shown that such a machine would provide a significant increase in processing power over what is presently available.

A method has been shown which can be used to program the Content Addressable Array Parallel Processor to perform image convolutions simply and efficiently. Such a program, for a simple convolution, was shown which operates in ninety-eight microseconds. The time of the algorithm is independent of the size of the image and depends only upon the size of the mask and, for bit serial processing, upon the number of bits in the pixel and mask values. A formula was given for a worst case time estimate and a factor for estimating normal case time from this was discussed. It was also noted that the method could be applied to masks of other than square shapes.

We have also shown how the CAAPP may be used in the analysis of motion from image sequences. For certain applications, the parallelism provided by the CAAPP would make it possible to perform such analysis robustly and nearly at video frame rate.

The key feature of this design is its integration of associativity with array processing. The result does well what each of these architectures normally can do individually but additionally may be applied in a number of ways that can only be approached by the integrated combination. Thus it becomes possible to perform both low level (such as image convolutions) and high level (such as real-time LISP [7]) processing in the same machine.

The importance of using a conservative approach to development cannot be over emphasized. Too often designs for computers have been fielded which push technology too far. The result has usually been a single machine which is nearly impossible to keep running and too costly to be replicated. By keeping our design conservative, we hope to produce a machine that is both useable and replicable at a reasonable cost (on the same order as a mini-mainframe). This would then make it possible for other research facilities to have similar machines without the extra cost of development. The study of parallelism and its applications can then be advanced by providing the research community with a useable standardized parallel processor.

REFERENCES

[1] Foster, Caxton C., "Content Addressable Parallel Processors". Van Nostrand Reinhold, New York, 1976.

[2] C. Weems, "Life is a CAM-Array Old Chum", unpublished paper, January 1980.

[3] C. Weems, S. Levitan, and C. Foster, "Titanic: A VLSI-Based Content Addressable Parallel Array Processor". Proceedings of IEEE International Conference on Circuits and Computers, September 1982, pp.236-239.

[4] Prazdny, K. "Determining the Instantaneous Direction of Motion from Optical Flow Generated by a Curvilinearly Moving Observer". Proc. of the Pattern Recognition and Image Processing Conference. Dallas, Texas, August 1981, pp. 109-114.

[5] Lawton, D. T., Steenstrup, M. E., and Weems, C., "Determination of the Rotational and Translational Components of a Flow Field Using a Content Addressable Parallel Processor". COINS Technical Report, May, 1983.

[6] Lawton, D. T. and Weems, C. "Incorporating Content Addressable Array Processors into Computer Vision Systems". In preparation.

[7] Bonar, J. and Levitan, S. "Real-Time LISP Using Content Addressable Memory". 1981 International Conference on Parallel Processing, Bellaire, Michigan, August 25-28, 1981.