DTIC FILE COPY

AD-A215 982

IDA DOCUMENT D-649

HIGH LEVEL VISION AND PLANNING WORKSHOP PROCEEDINGS

Michael Bloom. *Editor*

August 1989

DTIC
ELFCTE
DEC 27 1989
S E D

Prepared for
Defense Advanced Research Projects Agency

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street. Alexandria. Virginia 22311-1772

IDA

89 12 26 096

DEFINITIONS
IDA publishes the following documents to report the results of its work.

Reports
Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports
Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers
Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents
IDA Documents are used for the convenience of the sponsors or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

AD NUMBER	****************************	E501178	****************************
FIELD 2:	FLD/GRP(S)	120900	
FIELD 3:	ENTRY CLASS.	U	
FIELD 4:	NTIS PRICE	HC MF	
FIELD 5:	SOURCE NAME	[INSTITUTE FOR DEFENSE ANALYSES ALEXANDRIA VA	
FIELD 6:	UNCLASS. TITLE	[H]IGH-[L]EVEL [V]ISION AND [P]LANNING [W]ORKSHOP [P]ROCEEDINGS.	
FIELD 7:	CLASS. TITLE		
FIELD 8:	TITLE CLASS.	U	
FIELD 9:	DESCRIPTIVE NOTE	[F]INAL REPT.,	
FIELD 10:	PERSONAL AUTHORS	[B]LOOM, [M]ICHAEL [I].	
FIELD 11:	REPORT DATE	AUG 89	
FIELD 12:	PAGINATION	251P	
FIELD 13:	PROCESSING LEVEL		
FIELD 14:	REPORT NUMBER	[IDA]-[D]-649	
FIELD 15:	CONTRACT NUMBER	[MDA]903-89-[C]-0003	
FIELD 16:	PROJECT NUMBER		
FIELD 17:	TASK NUMBER		
FIELD 18:	MONITOR ACRONYM	[IDA/HQ], [SBI	
FIELD 19:	MONITOR SERIES	89-034738, [AD]-[E]501 178	
FIELD 20:	REPORT CLASS	U	
FIELD 21:	SUPPLEMENTARY NOTE		
FIELD 22:	ALPHA LIMITATIONS	[A]VAILABILITY CONTROLLED BY [IDA], [A]TTN: [TIS], [A]LEXANDRIA, [VA]22311. [A]NNOUNCEMENT ONLY; DOCUMENT WILL BE MADE AVAILABLE FROM [DTIC]AFTER PROCESSING.	
FIELD 23:	DESCRIPTORS	*[ARTIFICIAL INTELLIGENCE], [WORKSHOPS], [UNITED STATES], [ISRAEL].	
FIELD 24:	DESCRIPTOR CLASS.	U	
FIELD 25:	IDENTIFIERS	[LPN]-[IDA]-[A]-116, [SBI]1, [F]ISCAL YEAR 1990, 3-[D V]ISION, [H]IGH [L]EVEL [V]ISION AND [P]LANNING, [M]ACHINE [V]ISION, [M]ANUFACTURING AND [S]CHEDULING.	
FIELD 26:	IDENTIFIER CLASS.	U	
FIELD 27:	ABSTRACT	[T]HE SLIDES, PAPERS, AND GRAPHIC ILLUSTRATIONS PRESENTED AT THE JOINT [U].[S]. [I]SRAELI WORKSHOP ON ARTIFICIAL INTELLIGENCE ARE PROVIDED IN THIS [I]NSTITUTE FOR [D]EFENSE [A]NALYSES DOCUMENT. [T]HIS DOCUMENT IS BASED ON A BROAD EXCHANGE OF IDEAS ABOUT CURRENT APPROACHES AND RESEARCH ISSUES IN THE AREAS OF DESIGN AUTOMATION AND AUTONOMOUS ROBOTIC SYSTEMS. [A]LIST OF PARTICIPANTS IS PROVIDED ALONG WITH APPLICABLE REFERENCES FOR INDIVIDUAL PAPERS.	
FIELD 28:	ABSTRACT CLASS.	@	
FIELD 29:	INITIAL INVENTORY		
FIELD 30:	ANNOTATION		
FIELD 31:	SPECIAL INDICATOR		
FIELD 32:	REGRADE CATEGORY		
FIELD 33:	LIMITATION CODES	1	
FIELD 34:	SOURCE SERIAL	[F	
FIELD 35:	SOURCE CODE	179350	
FIELD 36:	DOCUMENT LOCATION		
FIELD 37:	CLASSIFIED BY		
FIELD 38:	DECLASSIFY ON		
FIELD 39:	DOWNGRDE TO CONF ON		
FIELD 40:	GEOPOLITICAL CODE	5108	
FIELD 41:	TYPE CODE	W	
FIELD 42:	IAC ACCESSION NO.	@@@P@@	
FIELD 43:	IAC DOCUMENT TYPE		
FIELD 44:	IAC SUBJECT TERM		
	...EXTENDED BY		

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE August 1989	3. REPORT TYPE AND DATES COVERED Final

4. TITLE AND SUBTITLE

High-Level Vision and Planning Workshop Proceedings

5. FUNDING NUMBERS

MDA 903 89 C 0003

A-116

6. AUTHOR(S)

Michael I. Bloom, Editor

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Institute for Defense Analyses
1801 N. Beauregard St.
Alexandria, VA 22311-1772

8. PERFORMING ORGANIZATION REPORT NUMBER

IDA Document D-649

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DARPA
1400 Wilson Blvd.
Arlington, VA 22209-2308

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release, unlimited distribution.

12b. DISTRIBUTION CODE

2A

13. ABSTRACT *(Maximum 200 words)*

The slides, papers, and graphic illustrations presented at the joint U.S.-Israeli workshop on artificial intelligence are provided in this Institute for Defense Analyses document. This document is based on a broad exchange of ideas about current approaches and research issues in the areas of design automation and autonomous robotic systems. A list of participants is provided along with applicable references for individual papers.

14. SUBJECT TERMS

Artificial Intelligence; Machine Vision; High Level Vision and Planning; 3-D Vision; Model-Based Object Recognition; Real-Time Planning; Reasoning with Interacting Goals; Manufacturing and Scheduling Problems.

15. NUMBER OF PAGES

251

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

IDA DOCUMENT D-649

HIGH LEVEL VISION AND PLANNING WORKSHOP PROCEEDINGS

Michael Bloom, *Editor*

August 1989

Accession For		
NTIS GRA&I		☒
DTIC TAB		☐
Unannounced		☐
Justification		
By		
Distribution/		
Availability Codes		
Dist	Avail and/or Special	
A-1		

IDA

INSTITUTE FOR DEFENSE ANALYSES

CONTENTS

Preface

JOINT US-ISRAELI WORKSHOP ON ARTIFICIAL INTELLIGENCE

This volume contains the Proceedings of a joint US-Israeli Workshop on Artificial Intelligence that was held the week of April 25, 1988 at the Weizmann Institute of Science, Rehovot, Israel, under the co-sponsorship of DARPA and the US-Israel Binational Science Foundation (BSF), and in conjunction with the Institute for Defense Analyses. The workshop brought together 9 American and 11 Israeli researchers, as well as 3 European scientists, to discuss and study selected problems in Artificial Intelligence, with emphasis on High Level vision and Planning. It provided an opportunity for a broad exchange of ideas about current approaches and research issues in these areas, as well as in related areas of design automation and autonomous robotic systems.

The workshop was hosted by the Israeli National Center for Artificial Intelligence, which was established in 1984 at the Weizmann Institute. The center is headed by Prof. Shimon Ullman - a distinguished researcher in the field of Computer vision who has a joint appointment at the Weizmann Institute and at MIT. Prof. Ullman was the Israeli coordinator of the Workshop. Dr. Saul Amarel, the previous Director of DARPA/ISTO, and LtCol Bob Simpson, ISTO Program Manager for Machine Intelligence, were US coordinators. Arrangement and editing of these proceedings was done by Michael Bloom of the Institute for Defense Analyses.

The idea of the joint workshop was proposed to DARPA by the Embassy of Israel in late '86. Subsequently, DARPA accepted the Israeli invitation to co-sponsor the workshop - thus endorsing the objective of promoting scientific cooperation between American and Israeli AI researchers, and with the added goal of contributing to progress in areas of AI science and technology that are of special interest to DARPA.

The technical program of the workshop included about twenty talks and one panel. Subjects discussed included computational studies of biological vision, algorithms for 3-D vision, model-based object recognition, design of integrated vision systems, control of search in planning, reasoning with interacting goals, real-time planning, and AI approaches to design, manufacturing and scheduling problems. The panel provided an opportunity to discuss the state of AI activities in Israel, and the state of AI in the US. In addition to the technical program, the Israeli hosts arranged tours to Massada, the Dead Sea and Jerusalem. In general, the local arrangements were outstanding.

The Israeli presentations covered academic as well as industrial research. AI is receiving increasing attention in Israel. Basic work in vision, and in other AI areas (including logic programming and parallel architectures for AI) is being conducted at the Weizmann Institute, the Technion in Haifa, the Hebrew University in Jerusalem, the Tel Aviv University, and the Ben Gurion University in the Negev. Research on vision and its military applications is carried out at RAFAEL (the Israel Armament Development Authority); and work on knowledge based, expert, systems in design, manufacturing and mission planning/control is underway in several industries, including the Israel Aircraft Industry (IAI).

The Israeli work which was presented at the workshop was state of the art and of high scientific quality. However, no major new ideas emerged in these presentations. Nevertheless, it appears that some of the Israeli research groups in AI have the potential of making significant contributions to the field. The chances of such contributions will increase if good scientific links

ii

are maintained with the American AI community. Conversely, the existence of such links will increase the chances that innovative concepts and applications that are developed in Israel will become readily available to US researchers. In view of these observations, the AI Workshop which was held at the Weizmann Institute can be considered to have been a success; it contributed substantially to the strengthening of the working links between US and Israeli AI researchers in areas of special importance to DARPA.

Robert L. Simpson, Jr.
Lieutenant Colonel, USAF
Program Manager for Machine Intelligence
Information Science and Technology Office
Defense Advanced Research Projects Agency

Saul Amarel
Alan M. Turing Professor of Computer Science
Rutgers University

PROGRAM
(*Papers included within)

Monday, April 25

Morning Session Chair: Professor S. Ullman

Welcome by A. Dvorstzky, President, The Weizman Institute

Computer Vision - The Challenge of a Multi Disciplinary Solution

Keynote Address: Z. Rotem, Executive Director, US-Israel Bi-National Science Foundation

The Growth and Mutual Benefit of the US-Israel Bi-National Cooperation in Science

*M. Fischler, *Machine Vision; Problems, Progress, Prognosis*

Y. Zeevi, *Cognitive Graph Approach to 3-D Vision*

J. Frisby, *Integration of Stereo, Texture, and Motion Cues in Human Slant Perception*

Afternoon Session Chair: Dr. T. Binford

S. Ullman. *Object Recognition by Alignment*

*J. Mayhew. *From TINA to ANIT: Towards as High Level Planning System*

O. Faugeras. *Shape Representation and Matching*

Tuesday, April 26

Morning Session Chair: R. Simpson

*R. Korf. *Planning as Search*

*J. Rosenschein. *Representation of Encounters Among Multiple Agents*

*S. Amarel. *Artificial Intelligence and Design: Opportunities, Challenges, Research Problems and Directions*

M. Ben-Bassat. *Large-Scale Timetable Scheduling*

Afternoon Session Chair: T. Kanade

M. Luria. *Knowledge Intensive Planner for the UNIX Consultant*

*S. Rosenschein. *New Foundations For Real-Time Perception-Action Systems*

THURSDAY, April 28

Morning Session Chair: O. Faugeras

*E.M. Riseman. *Overlapping Approaches In Perceptual Organization, Information Fusion, and Model-Directed Recognition*

T. Binford. *Generic Model Based Vision Interpretation*

*T. Kanade. *Towards Automatic Generation of Object Recognition Programs*

*Y. Yeshurun. *Space-Variant Vision: Implementation of Scanpath and Blending Algorithms for Contour-Based Scenes*

Afternoon Session Chair: S. Amarel

R. Shapira. *Vehicle Recognition*

G. Adiv. *3-D Motion and Image Matching From Line Correspondances*

*S. Peleg. *Characterization of Objects as Right-Handed or Left-Handed*

*J. Ben-Arie. *Using Simple Features for 3-D Object Recognition*

LIST OF PARTICIPANTS

Dr. G. Adiv
Rafael

Dr. S. Amarel
Rutgers University

Dr. G. Arieli
Ministry of Science, Israel

Dr. J. Ben-Arie
Technion

Dr. M. Ben-Bassat
Tel-Aviv University

Dr. T. O. Binford
Stanford University

Dr. A. Bruckstein
Technion

Dr. I. Dinstein
Ben-Gurion University

Dr. O. D. Faugeras
Domaine De Vol.-Rocq.

Dr. M. A. Fischler
SRI International

Prof. J. Frisby
University of Sheffield

Mr. H. Ganzer
Ministry of Defense, Israel

Dr. I. Inbar
Technion

Dr. T. Kanade
Carnegie-Mellon University

Dr. R. Korf
U.C.L.A.

Dr. M. Luria
Technion

Dr. J. Mayhew
University of Sheffield

Dr. S. Peleg
Jerusalem University

Dr. E. M. Riseman
University of Massachusetts

Dr. A. Rosenfeld
University of Maryland

Dr. J. Rosenschein
Jerusalem University

Dr. S. Rosenschein
SRI International

Dr. Z. Rotem
Binational

Dr. S. Shapira
Rafael

Dr. R. Simpson
Defense Advanced Research
Projects Agency, United States

Dr. S. Ullman
Weizmann Institute

Dr. Y. Yeshurun
Tel-Aviv University

Dr. Z. Zeevi
Technion

MACHINE VISION: PROBLEMS, PROGRESS, PROGNOSIS

By: Martin A. Fischler
Program Director, Perception
Artificial Intelligence Center

MACHINE VISION

Modeling the environment from sensor data and stored knowledge.

1. Recovering Scene Geometry

2. Detection and Delineation of Coherent Scene Components

3. Semantic Interpretation (assignment of names and labels)

RECOVERING SCENE GEOMETRY

- Conventional Approach

  - Two images

  - Local matching (correlation)

  - Critical assumptions

    * No occlusion

    * Identical appearance

    * Smooth surfaces
      (image patch at uniform depth)

- Progress

  - Global optimization (ambiguity, resolution)

  - Continuous viewing (matching, occlusion)

  - Visualization (evaluation, communication)

- Open Problem

  - Geometric recovery from a single image

IMAGE PARTITIONING

- Conventional Approach

    – Homogenity of local photometric image attributes (i.e., intensity, color, texture)

    – Continuity of local geometric structure (e.g., depth, contour)

- Progress

    – Best description (ability to deal with semantic content; subjective completion)

- Open Problems

    – Language and criteria for duplicating human performance

    – Indexing problem

4

- Conventional Approach

  - Classification of feature vectors of local attributes (statistical decision theory)

  - Matching of image structures to explicit geometric models (e.g., correlation)

- Progress

  - New description languages and associated computational procedures for matching and modeling classes of objects

- Open Problems

  - Frame problem

  - Recognition based on function, purpose, and context

OPEN PROBLEMS

- Geometric recovery from a single image

- Perceptual organization:

  - Language and criteria for first description (representation problem) suitable for indexing (frame problem)

- Recognition in the absence of strong models

HISTORICAL PERSPECTIVE

- Statistical decision theory (based on local image attributes)

- Physical and geometrical modeling (study of constraints imposed by physical world and imaging process)

- Global optimization (best description of image appearance with respect to a given language)

- Semantic interpretation (invoking knowledge of purpose, function, context)

TINA: The Sheffield AIVRU vision system.

J Porrill, SB Pollard, TP Pridmore, JB Bowen, JEW Mayhew & JP Frisby

AI Vision Research Unit
Sheffield University
Sheffield S10 2TN
England

Abstract

We describe the Sheffield AIVRU 3D vision system for robotics. The system currently supports model based object recognition and location; its potential for robotics applications is demonstrated by its guidance of a UMI robot arm in a pick and place task. The system comprises:

1) The recovery of a sparse depth map using edge based passive stereo triangulation.

2) The grouping, description and segmentation of edge segments to recover a 3D description of the scene geometry in terms of straight lines and circular arcs.

3) The statistical combination of 3D descriptions for the purpose of object model creation from multiple stereo views, and the propagation of constraints for within view refinement.

4) The matching of 3D wireframe models to 3D scene descriptions, to recover an initial estimate of their position and orientation.

Introduction.

The following is a brief description of the system. Edge based binocular stereo is used to recover a depth map of the scene from which a geometrical description comprising straight lines and circular arcs is computed. Scene to scene matching and statistical combination allows multiple stereo views to be combined into more complete scene descriptions with obvious application to autonomous navigation and path planning. Here we show how a number of views of an object can be integrated to form a useful visual model, which may subsequently be used to identify the object in a cluttered scene. The resulting position and attitude information is used to guide the robot arm. Figure 1 illustrates the system in operation.

The system is a continuing research project: the scene description is currently being augmented with surface geometry and topological information. We are also exploring the use of predictive feed forward to quicken the stereo algorithm. The remainder of the paper will
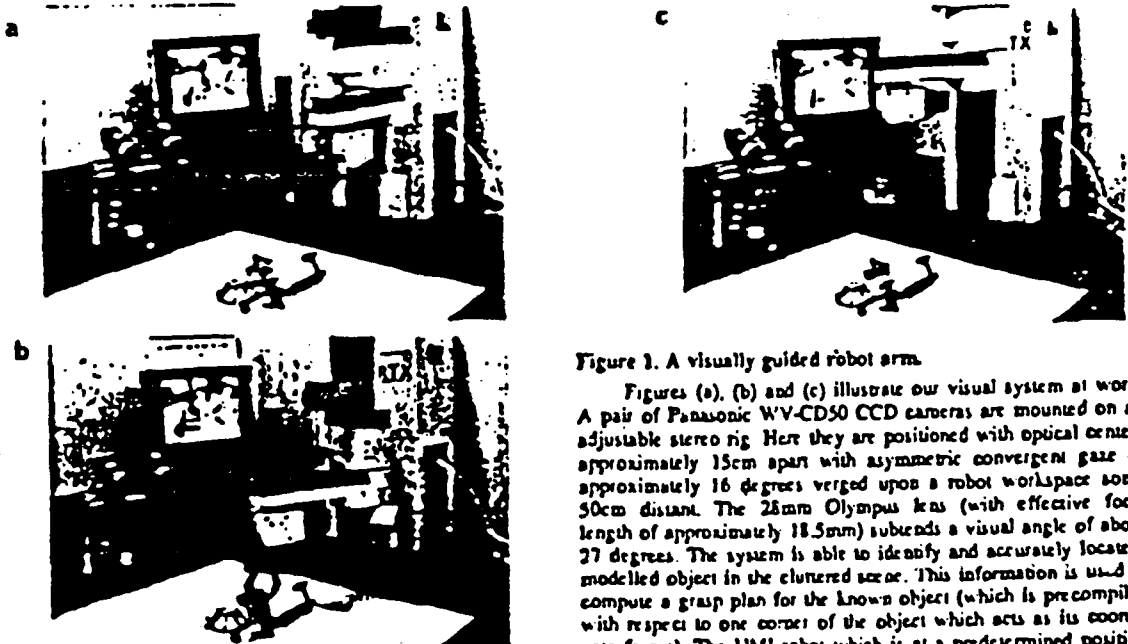
Figure 1. A visually guided robot arm.

Figures (a), (b) and (c) illustrate our visual system at work. A pair of Panasonic WV-CD50 CCD cameras are mounted on an adjustable stereo rig. Here they are positioned with optical centers approximately 15cm apart with asymmetric convergent gaze of approximately 16 degrees verged upon a robot workspace some 50cm distant. The 28mm Olympus lens (with effective focal length of approximately 18.5mm) subtends a visual angle of about 27 degrees. The system is able to identify and accurately locate a modelled object in the cluttered scene. This information is used to compute a grasp plan for the known object (which is precompiled with respect to one corner of the object which acts as its coordinate frame). The UMI robot which is at a predetermined position with respect to the viewer centered coordinates of the visual system is able to pick up the object.

8

describe the modules comprising the system in more detail.

PMF: The recovery of a depth map.

The basis is a fairly complete implementation of a single scale Canny edge operator [Canny 1983] incorporating sub pixel acuity (achieved through quadratic interpolation of the peak) and thresholding with hysteresis

applied to two images obtained from CCD cameras. The two edge maps are then transformed into a parallel camera geometry and stereoscopically combined (see figures 2, 3 and 4). The PMF stereo algorithm, described in more detail elsewhere [Pollard et al 1985; Pollard 1985], uses the disparity gradient constraint to solve the stereo correspondence problem. The parallel camera geometry allows potential matches to be restricted to corresponding rasters. Initial matches are further restricted to edge seg-
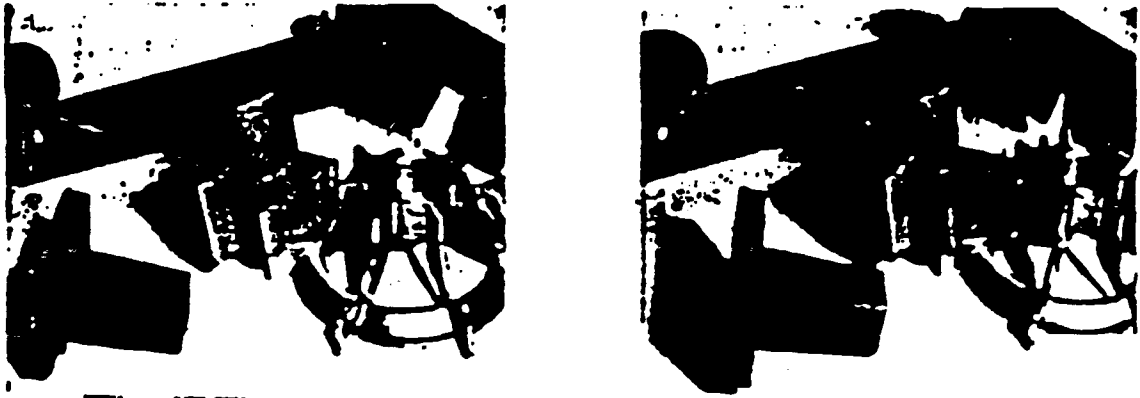

Figure 2. Stereo Images.

The images are 256x256 with 8 bit grey level resolution. In the camera calibration stage, a planar tile containing 16 squares equally spaced in a square grid was accurately placed in the workspace at a position specified with respect to the robot coordinate system such that the orientation of the grid corresponded to the XY axes. The position of the corners on the calibration stimulus were measured to within 15 microns using a Sielo 1818 stereo comparator. Tsai's calibration method was used to calibrate each camera separately. We have found errors of the same order as Tsai reported and sufficient for the purposes of stereo matching. The camera attitudes are used to transform the edge data into parallel camera geometry to facilitate the stereo matching process. To recover the world to camera transform the calibration images are themselves used as input to the system, eg are stereoscopically fused and the geometrical description of the edges and vertices of the squares statistically combined. The best fitting plane, the directions of the orientations of the lines of the grid corresponding to the XY axes, and the point of their intersection gives the direction cosines and position of the origin of the robot coordinate system in the camera coordinate system. The use of the geometrical descriptions recovered from stereo as feedback to iterate over the estimates of the camera parameters is a project for the future.
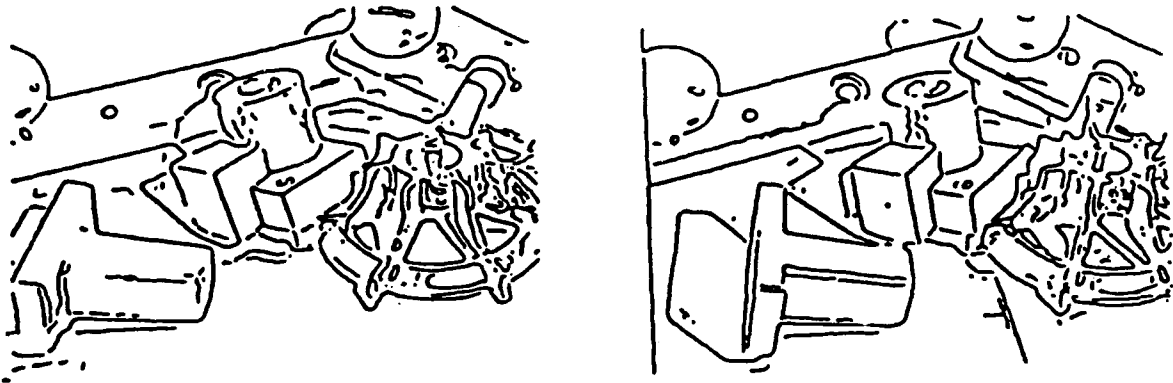
Figure 3. The edge maps.

A single scale Canny operator with sigma 1 pixel is used. The non maxima suppression which employs quadratic interpolation gives a resolution of 0.1 of a pixel (though dependent to some extent upon the structure of the image). After thresholding with hysteresis (currently non adaptive), the edge segments are rectified so as to present parallel camera geometry to the stereo matching process. This also changes the location of the centre of the image appropriately, allows for the aspect ratio of the CCD array (fixing the vertical and stretching the horizontal) and adjusts the focal lengths to be consistent between views.

Figure 4. The depth map.

The output of the PMF stereo-algorithm displayed (with respect to the left image) with disparities coded by intensity (near-dark far-light). The total range of disparities in the scene was approximately 55 pixels from a search window of 90 pixels. PMF is a neighbourhood support algorithm and in this case the neighbourhood was 10 pixels radius. The disparity gradient parameter to PMF was 0.5. The iteration strategy used a conservative heuristic for the identification of correct matches, and their scores were frozen. This effectively removes them from succeeding iterations and reduces the computational cost of the algorithm as it converges to the solution. 5 iterations were sufficient.

ments of the same contrast polarity and of roughly similar orientations (determined by the choice of a disparity gradient limit). Matches for a neighbouring point may support a candidate match provided the disparity gradient between the two does not exceed a particular threshold. Essentially, the strategy is for each point to choose from among its candidate matches the one best supported by its neighbours.

The disparity gradient limit provides a parameter for controlling the disambiguating power of the algorithm. The theoretical maximum disparity gradient is 2.0 (along the epipolars), but at such a value the disambiguating power of the constraint is negligible. False matches frequently receive as much support as their correct counterparts. However, as the limit is reduced the effectiveness of the algorithm increases and below 1.0 (a value proposed as the psychophysical maximum disparity gradient by Burt and Julesz [1980]), we typically find that more than 90% of the matches are assigned correctly on a single pass of the algorithm. The reduction of the threshold to a value below the theoretical limit has little overhead in reduction of the complexity of the surfaces that can be fused until it is reduced close to the other end of the scale (a disparity gradient of 0.0 corresponds to fronto-parallel surfaces). In fact we find that a threshold disparity gradient of 0.5 is very powerful constraint for which less than 7% of surfaces (assuming uniform distribution over the gaussian sphere: following Arnold and Binford

[1980]) project with a maximum disparity gradient greater than 0.5 when the viewing distance is four times the interocular distance. With greater viewing distances, the proportion is even lower.

It has been shown [Trivedi and Lloyd 1985; Porrill 1985], that enforcing a disparity gradient ensures Lipschitz continuity on the disparity map. Such continuity is more general than and subsumes the more usual use of continuity assumptions in stereo.

The method used to calibrate the stereo cameras was based on that described by Tsai [1986] (using a single plane calibration target) which recovers the six extrinsic parameters (3 translation and 3 rotation) and the focal length of each camera. This method has the advantage that all except the latter are measured in a fashion that is independent of any radial lens distortion that may be present. The image origin, and aspect ratios of each camera had been recovered previously. The calibration target which was a tile of accurately measured black squares on a white background was positioned at a known location in the XY plane of the robot work space. After both cameras have been calibrated their relative geometry is calculated.

Whilst camera calibration provides the transformation from the viewer/camera to the world/robot coordinate spaces we have found it more accurate to recover the position of the world coordinate frame directly. Stereo matching of the calibration stimulus allows its position in space to be determined. A geometrical description of the position and orientation of the of the calibration target is obtained by statistically combining the stereo geometry of the edge descriptions and vertices. The process is described in Pollard and Porrill [1986].

GDB: The recovery of the geometric descriptive base.

In this section we briefly report the methods for segmenting and describing the edge based depth map to recover the 3D geometry of the scene in terms of straight lines and circular arcs. A complete description of the process can be found in Pridmore et al [1986] and Porrill et al [1986a].

The core process is an algorithm (GDF) which recursively attempts to describe, then smooth and segment, linked edge segments recovered from the stereo depth map. GDF is handed a list of edge elements by CONNECT [Pridmore et al 1985]. Orthogonal regression is used to classify the input string as a straight line, plane or space curve. If the edge list is not a statistically satisfactory straight line but does form an acceptable plane curve, the algorithm attempts to fit a circle. If this fails, the curve is smoothed and segmented at the extrema of curvature and curvature difference. The algorithm is then applied recursively to the segmented parts of the curve.

Some subtlety is required when computing geometrical descriptions of stereo acquired data. This arises in part from the transformation between the geometry in disparity coordinates and the camera/world coordinates. The former is in a basis defined by the X coordinates in the left and

right images and the common vertical Y coordinate, the latter, for practical considerations (eg there is no corresponding average or cyclopean image), is with respect to the left imaging device, the optical centre of the camera being at (0,0,0) and the centre of the image is at (0,0,f) where f is the focal length of the camera. While the transformation between disparity space and the world is projective, and hence preserves lines and planes, circles in the world have a less simple description in disparity space. The strategy employed to deal with circles is basically as follows: given a string of edge segments in disparity space, our program will only attempt to fit a circle if it has already passed the test for planarity, and the string is then replaced by its projection into this plane. Three well chosen points are projected into the world/camera coordinate frame and a circle hypothesised, which then predicts an ellipse lying in the plane in disparity space. The mean square errors of the points from this ellipse combined with those from the plane provide a measure of the goodness of fit. In practice, rather than change coordinates to work in the plane of the ellipse, we work entirely in the left eye's image, but change the metric so that it measures distances as they would be in the plane of the ellipse.

Typically, stereo depth data are not complete; some sections of continuous edge segments in the left image may not be matched in the right due to image noise or partial occlusion. Furthermore disparity values tend to be erroneous for extended horizontal or near horizontal segments of curves. It is well known that the stereo data associated with horizontal edge segments is very unreliable, though of course the image plane information is no less usable than for the other orientations. Our solution to these problems is to use 3D descriptions to predict 2D data. Residual components derived from reliable 3D data
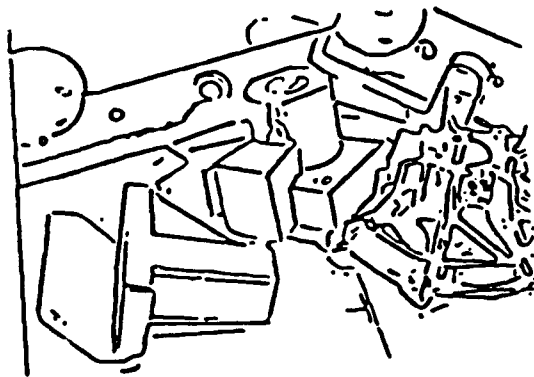

Figure 5. The geometric description overlaid on the left edge map.

The thin lines depict connected edge segments to which either no description has been ascribed because they were too short, or because they are present only in the left eyes image and only a 2D description was possible. The thicker lines depict the connected edge segments for which a 3D geometrical description has been computed. Before segmentation each edge list was smoothed by diffusion (see Porrill [1986]) approximately equal to a gaussian of sigma 2.5.

and the image projection of unreliable or unmatched (2D) edges are then statistically combined and tested for acceptance. By this method we obtain a more complete 2D and 3D geometrical description of the scene from the left eyes view than if we used only the stereo data. Figure 5 illustrates the GDB description of our scene.

Evaluation of the geometrical accuracy of the descriptions returned by the GDF has employed both natural and CAD graphics generated images. The latter were subject to quantisation error and noise due to the illumination model but had near perfect camera geometry; they were thus used to provide the control condition, enabling us to decouple the errors due to the camera calibration stage of the process. A full description of the experiments are to be found in Pridmore [1987], suffice it to say that we find that typical errors for the orientation of lines is less than a degree, and for the normals of circular arcs subtending more than a radian, the errors are less than 3 degrees in the CAD generated images and only about twice that for images acquired from natural scene. The positional accuracy of features and curvature segmentation points has also been evaluated, errors are typically of the order of a few millimetres which maybe argues well for the adequacy of Tsai's camera calibration method more than anything else.

SMM: The Scene and Model Matcher.

The matching algorithm (see Pollard et al [1986] for details), which can be used for scene to scene and model to scene matching, exploits ideas from several sources: the use of a pairwise geometrical relationships table as the object model from Grimson and Lozano-Perez [1984; 1985], the least squares computation of transformations by exploiting the quaternion representation for rotations from Faugeraus et al [1984; 1985], and the use of focus features from Bolles et al [1983]. We like to think that the whole is greater than the sum of its parts!
The matching strategy proceeds as follows:

1) a focus feature is chosen from the model;

2) the S closest salient features are identified (currently salient means lines with length greater than L);

3) potential matches for the focus feature are selected;

4) consistent matches, in terms of a number of pairwise geometrical relationships, for each of the neighbouring features are located;

5) the set of matches (including the set of focus features) is searched for maximally consistent cliques of cardinality at least C, each of these can be thought of as an implicit transformation.

6) synonymous cliques (that represent the same implicit transformation) are merged and then each clique is extended by adding new matches for all other lines in the scene if they are consistent with each of the matches in the clique. Rare inconsistency amongst an extended clique is dealt with by a final economical tree search.

extended cliques are ranked on the basis of the number and length of their members.

8) the transformation implicitly defined by the clique is recovered using the method described by Faugeras et al [1984].

The use of the parameters S (the neighbours of the focus feature), and C (the minimum subset of S) are powerful search pruning heuristics that are obviously model dependent. Work is currently in hand to extend the matcher with a richer semantics of features and their pairwise geometrical relationships, and also to exploit negative or incompatible information in order to reduce the likelihood of false positive matches.

TIED: the integration of edge descriptions.

The geometrical information recovered from the stereo system described above is uncertain and error prone, however the errors are highly anisotropic, being much greater in depth than in the image plane. This anisotropy can be exploited if information from different but approximately known positions is available, as the statistical combination of the data from the two viewpoints provides improved location in depth. From a single stereo view the uncertainity can only be improved by exploiting geometrical constraints. A method for the optimal combination of geometry from multiple sensors based on the work of Faugeras et al [1986] and Durrant-Whyte [1985] has been developed (for details see Porrill et al. [1986b]), and extended to deal both with the specific geometrical primitives recovered by the GDF and the enforcing of constraints between them. The method is used in the application being described to integrate the edge geometry from multiple views to create the object model (see figure 6), and to obtain the statistically optimum estimate of the position and direction cosines of the target object coordinate frame after the matching stage has been completed. The latter is done by enforcing the constraints that the axes of the coordinate frame are parallel to all the lines they should be, that they are mutually perpendicular, and intersect at a single point The result of the application of this stage of the process is the position and attitude of the object in the world coordinates. Figure 7 illustrates the SMM matching the compiled visual model in the scene. The information provided by matching gives the RHS of the inverse kinematics equation which must be solved if our manipulator is to grasp the object (see figure 8).

REV: the regions, edges, vertices graph.

One may regard the system as generating a sequence of representations each spatially registered with respect to a coordinate system based on the left eye: image, edge map, depth map and geometrical description. In the initial stages of processing a pass oriented approach may be appropriate but we consider that it is desirable to provide easy and convenient access between the representations at a higher level of processing. The REVgraph is an environment, built in Franz Lisp, in
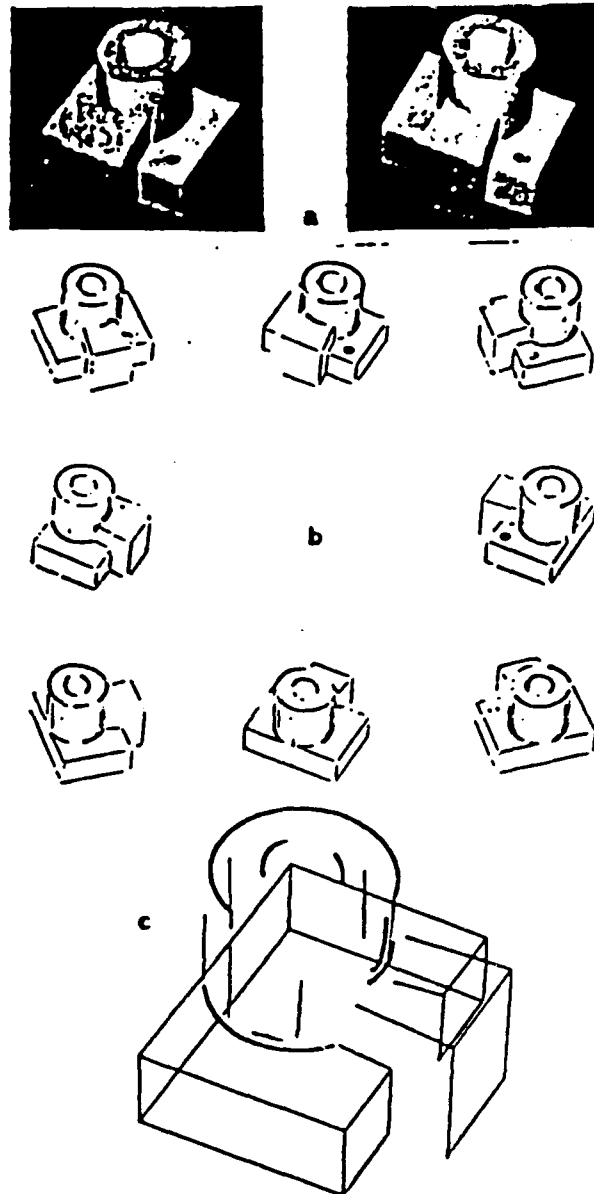
Figure 6. The integration of linear edge geometry from multiple views.

Figure (a) is a pair of stereo images produced by a version of the IBM WINSOM CSG body modeler. It depicts the object to be modelled. To ensure a description of the model suitable for visual recognition and to allow greater generality (the same approach has been successfully applied to natural images of a real object) we combine geometrical data from multiple views of the object to produce a primitive visual model of it. Figure (b) illustrates the 3D data extracted from eight views of the object. Their combination is achieved by incrementally matching each view to the next. Between each view the model is updated, novel features added and statistical estimation theory used to enforce consistency amongst them (eg. making near parallel and near perpendicular lines truely so). Finally only line features that have been identified in a more than a single view appear in the final visual model (see (c)).

12

a

Figure 7. Object location:

The dark lines depict the projection of the object model into the scene geometry after being transformed by the rotation and translation produced by the matching process (SMM) and the geometry integration process (TIED). The recovery of the object to scene transformation has two stages, they are as follows: first the matcher SMM locates the object model in the scene and recovers a sub-optimum estimate of the rotation and translation. The process is suboptimal because it does not take account of the anisotropies in the error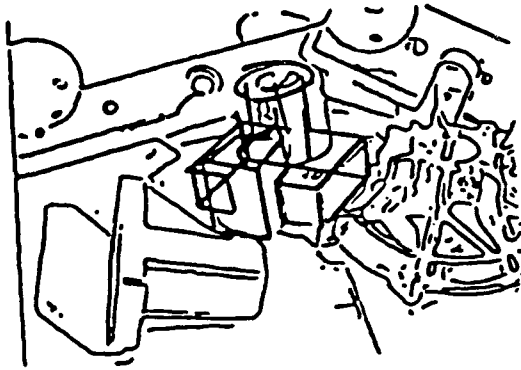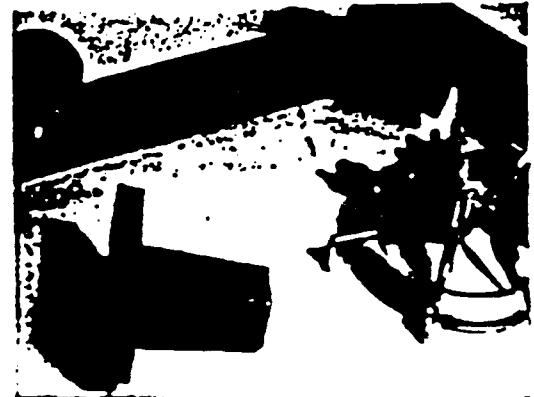s in the geometry of the matched edge features, and furthermore sequences the problem by first solving for the rotation and then using the rotation to calculate the translation. Notwithstanding these weaknesses, it is an adequate starting point for the second process which is a linearised recursive weighted least squares integration of the geometry (TIED), which delivers the corrected transformation. To give some idea of the scale of the matching search problem, the object model contains 41 features and the scene contains 423. Some 15 model focus features, chosen on the basis of length, resulted in the expansion of only 37 local cliques. The latter were required to be of magnitude at least C-4 from S-7 neighbouring features. The largest clique found by the matcher contained 14 matched lines.

b

Figure 8. Closing the loop.

Figures (a) and (b) show the arm grasping the object and the scene with the object removed.

which the lower level representations are all indexed in the same co-ordinate system. On top of this a number of tools have been and are being written for use in the development of higher level processes which we envisage overlaying the geometrical frame with surface and topological information. Such processes will employ both qualitative and quantitative geometrical reasoning heuristics. In order to aid debugging by keeping a history of reasoning, and increase search efficiency by avoiding backtracking, the REVgraph contains a consistency maintenance system (CMS), to which any processes may be easily interfaced. The CMS is our implementation of most of the good ideas in Doyle [1979] and DeKleer [1984] augmented with some our own. The importance of truth maintenance in building geometrical models of objects was originally highlighted by Hermann [1985]. Details of the REVgraph and CMS implementation may be found in Bowen [1986].

Conclusions

We demonstrate the ability of our system to support visual guided pick and place in a visually cluttered but, in terms of trajectory planning, benign manipulator workspace. It is not appropriate at this time to ask how long the visual processing stages of the demonstration take, suffice it to say that they deliver geometrical information of sufficient quality, not only for the task in hand but to serve as a starting point for the development of other visual and geometrical reasoning competences.

Acknowledgements

We gratefully acknowledge Dr Chris Brown for his valuable technical assistance.

References

Arnold R. D. and T. O. Binford (1980) Geometric constraints in stereo vision, Soc. Photo-Optical Instr. Engineers, 238, 281-292.

Bolles R.C., P. Horaud and M.J. Hannah (1983), 3DPO: A three dimensional part orientation system, Proc. IJCAI 8, Karlsruhe, West Germany, 116-120.

Bower J.B. and J.E.W. Mayhew (1986), Consistency maintenance in the REV graph environment, *Alvey Computer Vision and Image Interpretation Meeting*. University of Bristol, AIVRU Memo 20, and *Image and Vision Computing* (submitted).

Burt P. and B. Julesz (1980), Modifications of the classical notion of panum's fusional area, *Perception 9*, 671-682.

Canny J.F. (1983), Finding edges and lines in images, MIT AI memo, 720, 1983.

DeKleer J. (1984), Choices without backtracking, *Proc, National Conference on Artificial Intelligence*,

Doyle J. (1979), A truth maintenance system, *Artificial Intelligence* 12, 231-272.

Durrant-Whyte H.F. (1985), Consistent integration and propagation of disparate sensor observations, *Thesis, University of Pennsylvania*.

Faugeras O.D., M. Hebert, J. Ponce and E. Pauchon (1984), Object representation, identification, and positioning from range data, *Proc. 1st Int. Symp. on Robotics Res*, J.M. Brady and R. Paul (eds), MIT Press, 425-446.

Faugeras O.D. and M. Hebert (1985), The representation, recognition and positioning of 3D shapes from range data, *Int. J. Robotics Res*

Faugeras O.D., N. Ayache and B. Faverjon (1986), Building visual maps by combining noisy stereo measurements, *IEEE Robotics conference*, San Francisco.

Grimson W.E.L. and T. Lozano-Perez (1984), Model based recognition from sparse range or tactile data, *Int. J. Robotics Res*. 3(3): 3-35.

Grimson W.E.L. and T. Lozano Perez (1985), Recognition and localisation of overlapping parts from sparse data in two and three dimensions, *Proc IEEE Int. Conf. on Robotics and Automation*. Silver Spring: IEEE Computer Society Press, 61-66.

Grimson W.E.L. and T. Lozano-Perez (1985), Search and sensing strategies for recognition and localization of two and three dimensional objects, *Proc. Third Int. Symp. on Robotics Res*.

Herman M. (1985), Representation and incremental construction of a three-dimensional scene model, CMU-CS-85-103, Dept. of Computer Science, Carnegie-Mellon University.

Pollard S.B., J.E.W. Mayhew and J.P. Frisby (1985), PMF: a stereo correspondence algorithm using a disparity gradient limit, *Perception*, 14, 449-470.

Pollard S.B., J. Porrill, J.E.W. Mayhew and J.P. Frisby (1985), Disparity gradient, Lipschitz continuity and computing binocular correspondences, *Proc. Third Int. Symp. on Robotics Res*.

Pollard S.B., J.Porrill, J.E.W. Mayhew and J.P. Frisby (1986), matching geometrical descriptions in 3-space, *Alvey Computer Vision and Image Interpretation Meeting*. Bristol, AIVRU Memo 022 and *Image and Vision Computing (in press)*.

Pollard S.B. (1985), *Identifying Correspondences in binocular stereo*, unpublished Phd thesis, Dept of Psychology, University of Sheffield.

Pollard S.B. and J. Porrill (1986), Using camera calibration techniques to obtain a viewer centred coordinate frame, AIVRU Lab Memo 026, University of Sheffield.

Porrill J. (1985) Notes on: the role of the disparity gradient in stereo vision, AIVRU Lab Memo 009, University of Sheffield.

Porrill J., T. P. Pridmore, J. E. W. Mayhew and Frisby, J. P. (1986a) Fitting planes, lines and circles to stereo disparity data, AIVRU memo 017

Porrill J., S.B. Pollard and J.E.W Mayhew (1986b), The optimal combination of multiple sensors including stereo vision, *Alvey Computer Vision and Image Interpretation Meeting*. Bristol, AIVRU Memo 25 and *Image and Vision Computing (in press)*.

Pridmore T.P., J.E.W. Mayhew and J.P. Frisby (1985), Production rules for grouping edge-based disparity Data, *Alvey Vision Conference*, University of Sussex, and AIVRU memo 015, University of Sheffield.

Pridmore T.P. (1987), Forthcoming Phd Thesis, University of Sheffield.

Pridmore T.P., J. Porrill and J.E.W. Mayhew (1986), Segmentation and description of binocularly viewed contours, *Alvey Computer Vision and Image Interpretation Meeting*, University of Bristol, and *Image and Vision Computing (in press)*

Trivedi H.P. and S.A. Lloyd (1985), The role of disparity gradient in stereo vision, Comp Sys. Memo 165, GEC Hirst Research Centre, Wembley, England.

Tsai R.Y. (1986), An efficient and accurate camera calibration technique for 3D machine vision, *Proc IEEE CVPR 86*, 364-374.

Planning as Heuristic Search[*]

Richard E. Korf
Computer Science Department
University of California, Los Angeles
Los Angeles, Ca. 90024

June 14, 1988

Abstract

We propose the thesis that heuristic search is an effective paradigm for planning in certain domains. For example, we argue that a standard chess program is engaged in a form of multi-agent planning under real-time constraints. We then adopt the standard two-player game algorithms to single-agent problems. We develop a special case of minimax search along with a powerful pruning algorithm analogous to alpha-beta. Real-Time-A* is a generalization of A* that makes moves in constant time and allows backtracking while guaranteeing finding a solution. The algorithm effectively solves larger problems than have previously been solved with heuristic search techniques.

1 Introduction

1.1 What is Planning?

The term *planning* is often used in the artificial intelligence literature, but without a precise, generally agreed-upon meaning. A large body of work in the 1970's, associated in particular with the blocks world domain, has been

called planning. That effort was focussed on problem solving in the presence of interacting subgoals and abstract problem spaces. Subsequently, much of that work has been recast as search in a problem space, but using subgoals and abstraction spaces as knowledge sources, as opposed to heuristic evaluation functions [6].

An implicit assumption of that work was that the preconditions and effects of actions in the world could be completely and simply specified. More recently, a large number of researchers have challenged those assumptions, striving for more robust and formal theories of action and time. For example, two related problems of interest to this community are the *qualification* and *ramification* problems. The qualification problem is how to predict whether an action will succeed when there can potentially be an infinite number of preconditions to that action. For example, if I turn the key in my car's ignition, the car will start unless it is out of gas, or the battery has been stolen, or there is a potato in the tailpipe, etc. Similarly, the ramification problem is how to predict the effects of an action when they too can be infinite in number. For example, when I turn the key in the ignition, the car may start, or it may be in gear and crash through the garage, or it may explode, etc. This general body of work on reasoning about action and time is also referred to as planning. A recent example of this type of work can be found in [2,3].

Yet a third view of planning is the non-technical or layman's view of the term. Perhaps the most common example is planning a trip, involving such things as air travel, ground transportation, hotel accomodations, etc. The essence of this activity is to create a symbolic data structure, called a *plan*, that specifies, at a certain level of detail, the actions that we anticipate performing when we actually execute the plan or take the trip. The reason we do this is to anticipate problems before they actually occur. For example, it is much less costly to realize from an airline schedule that we won't be able to make a connecting flight, than to actually fly the first leg and then realize that we missed the second.

In fact, all three of these views are consistent and compatible. They all involve simulating single actions or sequences of actions in order to predict if they will achieve a goal. Of course the prediction cannot be perfect, and ultimately the planned actions must actually be executed in the real world and their effects monitored. When unanticipated results occur, replanning is often necessary to get from the new state to the goal.

In addition, planning occurs at many levels of detail, and at different times relative to execution. For example, the actual flights taken on a trip may be booked months in advance, while transportation to the airport may only be arranged a few days ahead of time, and the path traversed in walking to the gate is decided at execution time.

1.2 Playing Chess as Planning

It is generally believed that planning research in AI is in its infancy and has yet to be implemented and used in real systems. Given our characterization of planning above, however, planning is ubiquitous in AI systems, but simply not recognized as such. As an example, we will consider a standard chess program based on heuristic search as a planning system. This point of view was mentioned by Rolf Stachowitz at a recent workshop on planning held in Santa Cruz, Ca., in October, 1987.

A chess program expands the game tree to some fixed search depth, and evaluates each of the terminal positions according to a static evaluation function. It then backs up the frontier values using the minimax rule, augmented by alpha-beta pruning. The result of this is a *strategy* for the player to move. A strategy for a player is a subtree of the complete game tree that contains the root node, one child of every node where the player is to move, and all children of every node where the opponent is to move, up to the search horizon. A strategy is a plan. It is a data structure that specifies the best move for the player, contingent on each possible move the opponent could make. Finally, the program executes the first move of its strategy. In general, a new plan is computed for each move, but some programs save some of the previous plan instead of recomputing it from scratch.

Thus, a chess program is engaged in an elaborate planning process, often extending eight ply deep. While this behavior is usually called heuristic search instead of planning, it meets our definition of planning, namely simulating action to decide what operation to execute next. It performs this planning under real-time constraints, since in a tournament setting moves must be made in a fixed constant time. Furthermore, this planning is done in the context of tightly coupled interactions with another agent, the opponent. Finally, the resulting behavior exhibits truly expert performance. The best of current chess machines outperform 99.5% of rated human tour-

nament players [1]. The question we address in the remainder of this paper is to what extent this paradigm can be adopted to single-agent planning problems.

2 Real-Time Single-Agent Search

Research on two-player games has always assumed insufficient computational power to search all the way to terminal positions, and that moves must be irrevocably committed under strict time constraints. Conversely, research on single-agent problems has usually assumed that search can proceed to goal positions, that an entire solution may be computed before even the first move need be executed, and that optimal solutions are required. As a result, existing single-agent heuristic search algorithms, such as A* [4] and IDA* [5], do not scale up to large problems due to their exponential complexity, a necessary consequence of finding optimal solutions. This work extends the techniques of heuristic search to handle single-agent problems under conditions of limited computation where decisions must be committed to in constant time per move. A key step is to give up solution optimality, and to assume that computational resources do not permit searching all the way from the initial state to a goal state.

2.1 Minimin Lookahead Search

The obvious first step is to specialize minimax search to the case where a single-agent makes all the moves. The resulting algorithm, called minimin search, searches forward from the current state to a fixed depth horizon determined by the computational resources available, and then applies the A* cost function of $f(n) = g(n) + h(n)$ to the frontier nodes. Since a single agent makes all the decisions, the minimum value is then backed up, instead of the minimax value, and a single move is made in the direction of the minimum value. Making only a single move at a time follows a strategy of least commitment, since the backed-up values are only heuristic, and further search may recommend a different second move.

2.2 Alpha Pruning

There exists an analog to alpha-beta pruning that makes the same decisions as full minimin search, but by searching fewer nodes. It is based on the assumption that the $f = g + h$ cost function is monotonically non-decreasing along any path. Since this condition is equivalent to h being a metric, and by definition all reasonable cost functions are metrics, the monotonicity condition is not a restriction in practice. Given monotonicity, and static evaluations of all interior nodes, branch-and-bound can applied as follows. The value of the best frontier node encountered so far is stored in a variable called α, and whenever the cost of a node equals or exceeds α, the corresponding branch is pruned off. The reason is that all descendents of that node must have costs that are greater than or equal to α. In addition, whenever a frontier node is encountered with a value less than α, α is reset to this lower value.

The performance improvement due to alpha pruning is quite dramatic. In some cases, it extends the achievable search horizon by a factor of five, for a fixed amount of computation. Even more surprising is the fact that the search horizon reachable with this algorithm *increases* with increasing branching factor! The reason is that with a larger branching factor, lower values of α are achieved earlier in the search, resulting in greater savings through pruning.

Minimin lookahead search with alpha pruning is a strategy for evaluating the immediate children of the current node. It constitutes the planning phase where the moves are merely simulated, rather than being executed in the real world. This is completely analogous to minimax search with alpha-beta pruning. As such, it can be viewed as providing a range of more accurate but computationally more expensive heuristic functions, one corresponding to each search horizon.

2.3 Real-Time-A*

The next step is to control the sequences of moves actually executed. In the two-player game setting, this problem is trivial since the minimax algorithm is simply repeated for each move. In a single-agent problem, however, this naive strategy will often lead to infinite loops. In addition, in a single-agent problem, backtracking may be possible. What is required is an algorithm

that permits backtracking when it appears favorable in light of additional information, but prevents infinite loops and guarantees that a solution will be found if it exists.

Real-Time-A* (RTA*) is such an algorithm. The basic idea is that the current path should be abandoned in favor of a previous path when the estimate of completing the current path exceeds the estimate for a previous path plus the cost of backtracking to that path. This can be achieved by modifying the definition of $g(n)$ in A* to be the distance to node n from the current state of the problem solver, rather than from the initial state. Unfortunately, this would require updating the value of g for every node on OPEN with every move, and maintaining path information from the current state to every node on OPEN.

RTA*, however, implements this policy using only local information and control as follows: The neighbors of the current state are generated and a heuristic function, including lookahead search with alpha pruning, is applied to each new state. The neighbor with the minimum $g + h$ value is chosen as the new current state, and the old current state is stored in a table along with the node with the second best $g + h$ value. This represents the best estimate of the cost of finding the solution via the old current state from the perspective of the new current state. The algorithm simply repeats this cycle, using the stored h values for previously visited states, and computing it for new states, until a solution is found. It can be proven that this algorithm will always find a solution in any finite problem space in which there exists a path from every state to a goal, regardless of the initial values of the heuristic function.

3 Experimental Results

RTA* using minimin lookahead search with alpha pruning was implemented for various size sliding tile puzzles, using the Manhattan Distance heuristic function. It was tested on the Eight, Fifteen, and 5 × 5 Twenty-Four Puzzle, with search horizons ranging from one to 25 moves. As expected, the solution lengths decrease with increasing search horizon, with the largest improvements coming from the initial increases in search horizon. Solutions to the Fifteen Puzzle are found in about a second of CPU time, and solutions within a factor of two of optimal require only tens of seconds. The

Twenty-Four Puzzle, which has not previously been solvable with heuristic search techniques, also yields to this algorithm in a matter of seconds.

4 Conclusions

Heuristic search is a powerful paradigm for planning under real-time constraints, and provides a natural framework for the interleaving of planning and execution. We have adopted the standard two-player game algorithms, and developed a new algorithm (RTA*) that increases the size of single-agent problems that can be effectively solved using heuristic search. Some of this work has appeared in [7,8]. A more complete treatment can be found in [9]. The question of how large a problem these techniques scale up to is the subject of current research.

References

[1] Berliner, H., and C. Ebeling, Pattern knowledge and search: The SUPREM architecture, Technical report CMU-CS-88-109, Computer Science Department, Carnegie-Mellon University, Pittsburgh, Pa., Jan. 1988.

[2] Ginsberg, M.L., and D.E. Smith, Reasoning about Action I: A possible worlds approach, *Artificial Intelligence*, Vol. 35, No. 2, 1988, pp. 165-195.

[3] Ginsberg, M.L., and D.E. Smith, Reasoning about Action II: The qualification problem, *Artificial Intelligence*, Vol. 35, No. 3, 1988.

[4] Hart, P.E., N.J. Nilsson, and B. Raphael, A formal basis for the heuristic determination of minimum cost paths, *IEEE Transactions on Systems Science and Cybernetics*, SSC-4, No. 2, 1968, pp. 100-107.

[5] Korf, R.E., Depth-first iterative-deepening: An optimal admissible tree search, *Artificial Intelligence*, Vol. 27, No. 1, 1985, pp. 97-109.

[6] Korf, R.E., Planning as search: A quantitative approach, *Artificial Intelligence*, Vol. 33, No. 1, 1987, pp. 65-88.

[7] Korf, R.E., Real-time heuristic search: First results, *Proceedings of the National Conference on Artificial Intelligence (AAAI-87)*, Seattle, Wash., July, 1987, pp. 133-138.

[8] Korf, R.E., Real-time heuristic search: New results, *Proceedings of the National Conference on Artificial Intelligence (AAAI-88)*, Minneapolis, Mn., August, 1988.

[9] Korf, R.E., Real-time heuristic search, to appear, *Artificial Intelligence*, 1989.

Communication-Free Interactions Among Rational Agents:
A Probabilistic Approach

Jeffrey S. Rosenschein
Computer Science Department, Hebrew University
Givat Ram, Jerusalem, Israel
and
Rockwell Science Center, Palo Alto Laboratory, 444 High Street
Palo Alto, California 94301

John S. Breese
Rockwell Science Center, Palo Alto Laboratory, 444 High Street
Palo Alto, California 94301

January 12, 1989

Abstract

Recent work on interactions among rational agents has put forward a computationally tractable, deduction-based scheme for automated agents to use in analyzing multi-agent encounters. While the theory has defined irrational actions, it has underconstrained an agent's choices: there are many situations where an agent in the previous framework was faced with several potentially rational actions, and no way of choosing among them. This paper presents a probabilistic extension to the previous framework that provides agents with a mechanism for further refining their choice of rational moves. At the same time, it maintains the computational attractiveness of the previous approach.

The probabilistic extension is obtained by a representation of interactions that explicitly incorporates uncertainty about other players' moves. A three-level hierarchy of rationality is defined, corresponding to ordinal, stochastic, and utility dominance among alternative outcomes. The previous deduction-based formalism is recast in probabilistic terms, and is seen to be a particular special case of a more encompassing dominance theory. A technique is presented for using ordinal, stochastic, and utility dominance in interactions with other agents operating under various axioms of rationality.

1 Introduction

1.1 Interactions Among Rational Agents

Research on artificial intelligence (AI) has begun to concern itself with the design of an autonomous agent operating in real-world environments. Along one dimension, this requires that the agent be capable of dealing with dynamic and incompletely specified situations. It must be able to reason about change, recover from failures, and deal with uncertainty both in the state of the world and in the effects of its own actions.

Another capability of an autonomous agent that would be highly desirable in real-world settings would be its ability to interact flexibly with other agents. There are, in fact, few scenarios where an agent could be expected to operate with *complete* autonomy; almost always there will be others with whom the agent must interact. This will be true whether the agent is operating on a factory floor, building outposts on Mars, or running errands to the corner store. Furthermore, these other agents will in general possess a wide range of reasoning capabilities; the agent should be capable of interacting flexibly with agents of different rationality "types".

There has been considerable work in recent years by AI researchers on formalisms for representing inter-agent beliefs [12,13,1,14,15,7], an important component of the reasoning necessary for cooperation. Agents must reason about one another's beliefs to predict activity, provide information, and adapt their own behavior to others' expectations.

Another line of work has been considering the agent interactions themselves as objects about which to reason [4,5,17]. In this research, the agents

have been defined as operating under the constraints of various *rationality axioms* that restrict their choices in interactions. The effects of various axioms and their relationships to one another have been analyzed.

The current paper continues along this latter line of research. The basic extension proposed is to recognize that reasoning about other agents' actions must deal with uncertainty, and to incorporate an explicit mechanism for doing so. Uncertainty is inherent in encounters because of incomplete information about others' objectives, options, and reasoning processes. Our addressing of uncertainty issues comes against the backdrop of an increased use of the decision-theoretic concepts of probability and utility theory in AI research [3,8]. At the same time, we exploit the fact that decision- and game-theorists have been considering the use of Bayesian decision theory in situations of strategic interaction [2,9,23,16]. Though we are not proposing an extension to the concepts of equilibrium proposed by game-theorists, the work in this paper integrates previous studies of rational interaction based on a deductive framework with decision-theoretic ideas and is a first step towards operationalizing recent advances in game-theoretic solution concepts.

1.2 Perspectives on Multi-Agent Interactions

We examine reasoning about other agents from two different perspectives, the "prescriptive/descriptive" approach and the "jointly prescriptive" approach. Both perspectives have their place in the theory of rational interacting agents, though each causes us to ask different questions about how automated agents should be designed. The bulk of this paper is focused on prescriptive/descriptive issues, though we make several observations and report results regarding jointly prescriptive methods.

1.2.1 Prescriptive/Descriptive

A "prescriptive/descriptive" approach requires two types of theories [10] to fully capture a multi-agent interaction. First, we need a normative theory of what our primary agent *should* do given his values and information. We have a prescriptive theory when we not only define these normative principles for rational behavior, but augment this with a prescription or method for identifying rational moves. This is precisely the approach we take in developing

our notion of prescriptive rationality. Second, we require a descriptive theory of other agents. A descriptive theory is useful to the extent it can be used to predict the actions of other agents, and may be based on varying degrees of assumed "rationality" of others.

The "prescriptive/descriptive" approach is basically decision analytic: using our model of interaction, we prescribe a particular course of action for one agent based on the description he has of other agents. This was the approach taken in previous DAI work such as [17], where different information about others' rationality would cause an agent to act appropriately. This "prescriptive/descriptive" perspective is central in our design of an agent capable of interacting intelligently, particularly when we will have no control over (and limited information about) the design of the other agents.

1.2.2 Jointly Prescriptive

Of course, if our descriptive theory is the same as our prescriptive theory, i.e., if the best theory one has about other agents is based on introspection regarding one's own reasoning processes, this results in a "jointly prescriptive" approach. "Jointly prescriptive" concerns form the basis for much of modern game theory [11]. These approaches, by and large, develop models of interaction that have certain globally desirable properties, given that all agents subscribe to the same fundamental solution strategies and have common knowledge regarding most aspects of the problem. The "jointly prescriptive" perspective is well-suited to closed systems where the interacting agents are all centrally designed. With total control over their methods of interaction (and hence the ability to engineer away uncertainty regarding others' decision-making strategies), the designer is looking for desirable properties, such as stability and pareto-optimality of solutions.

The jointly prescriptive perspective also has a role to play in competitive interactions. For example, some interaction strategies are known from the game theory literature to be "stable," i.e., if an agent uses this strategy, no opponent can benefit from playing any other strategy. A designer could feel safe in incorporating such a strategy into his agent—he need have no fear of the strategy's presence becoming known, since there is no effective counterstrategy. Thus the identification of stable strategies (which is a jointly prescriptive notion) can be important to any single agent's designer. Similarly,

a demonstration of a strategy's stability *and* pareto-optimality might be an effective argument in getting many agents' designers to incorporate it:[1] the best that other agents can do is to "play along," and the overall final results have certain desirable characteristics.

1.3 Assumptions

This paper is concerned with single interactions among agents: though there is a mechanism for using the results of past encounters, there is no explicit concern about future interactions. Each agent is assumed capable of assigning some value to a hypothetical outcome, and (in this paper) we will assume that these assigned payoff values, for all agents, are common knowledge among them all.[2] In addition, once the interaction has been recognized, there is no further communication among the agents; each must decide on its action alone. This is the no-communication scenario used in [5,4,17].[3] The agents are assumed to be operating under certain axioms, to be discussed, that control their behavior.

1.4 Overview

In Section 2 we introduce the formal notation for our analysis. In Section 3, various forms of dominance among alternatives are developed. The deduction-based formalism given in [17] is recast in probabilistic terms, and is seen to be a particular special case of a more encompassing dominance theory.

In Section 4 we consider questions relating to the design of an agent using the dominance relations, in the "prescriptive" portion of a "prescriptive/descriptive" approach. Axioms of behavior are given in Section 5 that

---

[1]This was, for example, the argument made in [6]

[2]Uncertainty about payoffs can be incorporated into the framework, and is topic for future research. Also, common knowledge is not always required; for a fuller discussion of how much knowledge is actually needed, see [18].

[3]While this scenario is a simplification of what might be found in real-world encounters, it is a useful starting point for an analysis of interactions. There are also a variety of instances when the assumption that no communication is possible is quite realistic, such as when agents designed in different countries or by different manufacturers unexpectedly encounter one another.

might describe our agent's opponents[4], and the ramifications these axioms have on the prescriptive dominance techniques are discussed. In Section 6 we briefly consider, from the "jointly prescriptive" perspective, the global properties of the methods we have outlined.

2   Notation

We will follow the convention of representing a game as a payoff matrix. Figure 1 is a representation of a two agent encounter.

$$\mathbf{K}$$

		c	d
a		3 ^{1}	1 ^{2}
b		2 ^{5}	0 ^{1}

$\mathbf{J}$

Figure 1: A Payoff Matrix

To a game corresponds a set P of players and, for each player $i \in P$, a set M_i of possible moves for i. For $S \subset P$, we denote $P - S$ by $\overline{S}$. We denote by m_S an element of M_S; this is a collective move (or a "joint" move) for the players in S. To $m_S \in M_S$ and $m_{\overline{S}} \in M_{\overline{S}}$ corresponds an element $\vec{m}$ of M_P. The payoff function for a game is a function

$$p : P \times M_P \to \mathbb{R}$$

whose value at $(i, \vec{m})$ is the payoff for player i if move $\vec{m}$ is made. The function p thus encodes the payoff matrix in function form.

We denote by $prob_i(m_{\overline{i}} \mid m_i, \xi)$ the probability distribution that agent i has over all the other players making move $m_{\overline{i}}$ (with ξ representing i's knowl-

---

[4]Throughout this paper, our use of the term "opponent" should not be taken in its colloquial sense. When our agent interacts with other agents, we will sometimes refer to them as its opponents, without intending that the agents are necessarily involved in conflict. There may be a convergence of interests among all parties.

edge of the world, including his knowledge of other agents). The probability may depend, as seen from this expression, on i's own move m_i.

We could use dual matrices to represent an interaction between agents, with associated probability distributions on their moves. Consider the two matrices in Figure 2.

K

$prob_K(m_J \mid m_K)$

		c	d
J	a	1 ⁴	3 ³
	b	2 ¹	4 ²

$prob_J(m_K \mid m_J)$

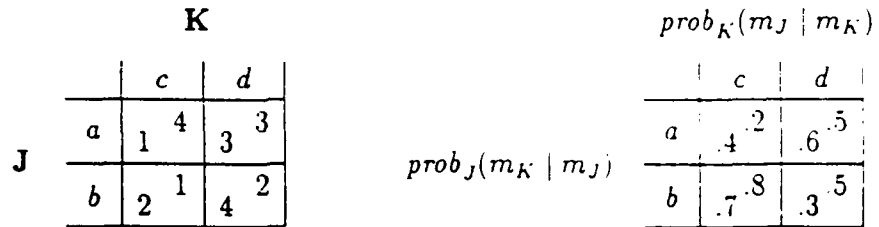|	c	d
a	.4 ·²	.6 ·⁵
b	.7 ·⁸	.3 ·⁵

Figure 2: Payoff and Probability Matrix

The left matrix is to be interpreted in the same manner as it was above, namely as defining the payoffs each agent will receive from various outcomes. In addition, each agent is assumed to have a probability distribution on the other's moves, given a move of his own. The second matrix in Figure 2 displays these distributions. For example, if J considers that he will make move b, he considers that there is a .7 probability that K will make move c, and a .3 probability that K will make move d. Of course, in the probability matrix the columns sum to 1 for K, and the rows sum to 1 for J.

We define a secondary payoff function $pay(i, m_i)$, which gives us the *set* of possible payoffs to i of making move m_i:

$$pay(i, m_i) = \{p(i, \vec{m}) : prob_i(m_{\bar{i}} \mid m_i, \xi) > 0\}. \tag{1}$$

The expression $prob_i(m_{\bar{i}} \mid m_i, \xi) > 0$ denotes the set of responses "considered possible" to i's move m_i.[5] There are many potential moves that might be expected of other agents, depending on assumptions about them (and their assumptions about you), and similarly, many different subjective probability distributions that one might have over their potential moves. In Section 5 we list several alternate definitions and indicate how each affects the *pay* function or probabilities.

---

[5] Careful readers will note that this expression subsumes the role of the *allowed* function in [17].

28

The final element of our notation that needs to be introduced is the notion of a "utility function" over payoffs. The utility function summarizes the agent's attitudes toward uncertain options, while payoffs summarize the agent's valuation under certainty of each possible joint move. The *utility of a joint move* for agent i in our notation is represented as $U_i(p(i,\vec{m}))$; it is a function from the real numbers to the real numbers. We then define the *expected utility* for agent i of a joint move as follows:

$$EU_i(\vec{m}) = \sum_{m_\tau \in M_\tau} U_i(p(i,\vec{m}))prob_i(m_\tau \mid m_i, \xi).\qquad(2)$$

More generally, the summation can be replaced by an integration. Von Neumann and Morgenstern, in their foundational work [22], formalized rationality in terms of axioms which require an agent to behave as if maximizing expected utility.

3 Dominance

A concept essential to this work is *dominance*: when the payoffs resulting from one move are better than those resulting from some other move, for some precise definition of "better," the inferior move is said to be *dominated*. Previous treatments such as [17] used only one kind of dominance, namely *ordinal dominance*, an "absolute" dominance between the members of two sets. In this paper we consider how two other kinds of dominance, *stochastic* and *utility* dominance, can be combined with the axiomatic approach.

3.1 Ordinal Dominance

Ordinal dominance is straightforward: for nonempty sets $\{\alpha_i\}$ and $\{\beta_j\}$, we say that $\{\alpha_i\}$ is ordinally dominated by $\{\beta_j\}$ (written $\{\alpha_i\} <_o \{\beta_j\}$) if $\alpha_i \le \beta_j$ for all i, j, and at least one element of $\{\alpha_i\}$ is less than every element of $\{\beta_j\}$. For example, the set $\{5,3\}$ ordinally dominates the set $\{3,1\}$, since *every* member of the first is greater than or equal to *every* member of the second, and in at least one case the relationship is strictly greater than.

3.2 Stochastic Dominance

3.2.1 The Intuition

Before launching into the formal definition of stochastic dominance, we will present the intuitions behind its use.

A *lottery* is defined to be a set of payoffs with associated probabilities. A lottery can be viewed is a state contingent payoff- in an interaction between agents the payoff is contingent on the (uncertain) move of the opponent.

Stochastic dominance [24] between two alternative lotteries is commonly represented graphically as follows. Consider a graph whose x-axis represents various payoffs, and whose y-axis represents cumulative probabilities (i.e., runs from 0 to 1). For each agent's lotteries, we draw a curve onto this coordinate space whose y position at any x value represents the probability that the agent will receive *less than* that value from that lottery. Each curve begins at the point $(p, 0)$ and increases to a maximum of $(q, 1)$, where p and q are the minimum and maximum possible payoffs, respectively. If one lottery's curve lies completely above and to the left of another lottery's curve (with possible overlap—but no crossing—of the curves permitted), we say that the first lottery is stochastically dominated by the second. This means that for any given value, the player has a better chance of getting it (or more) from the second lottery than from the first.

For example, consider an agent that has two lotteries available to him. In the first, he has .2 chance of getting a payoff of 4, a .5 chance of getting a payoff of 6, and a .3 chance of getting a payoff of 7. We draw this lottery's curve as in Figure 3.[6] The curve rises by .2 at 4, rises an additional .5 at 6, and rises an additional .3 at 7.

Now imagine that there is a second lottery, where he has a .3 chance of getting a payoff of 3, a .2 chance of getting 5, and a .5 chance of getting 6. This second lottery's curve looks like that in Figure 4.

If we now combine these two curves, it is evident that at all points the second curve are above those of the first curve for a given payoff—thus, the second lottery has a higher probability of getting a particular value or less for

---

[6]The diagram in Figure 3 is typical of lotteries with discrete moves and payoffs—a step function. We could, just as easily, have a continuous set of payoffs, which would result in a smooth curve in the diagram with no vertical climbs.

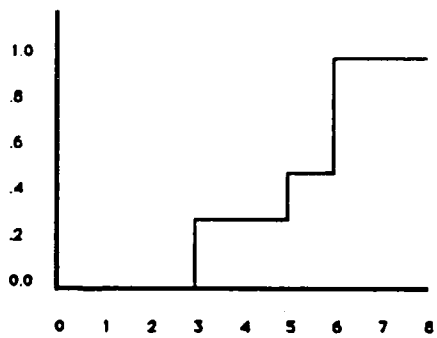Figure 3: Agent's First Lottery

Figure 4: Agent's Second Lottery

all values and therefore the first lottery stochastically dominates the second (see Figure 5).
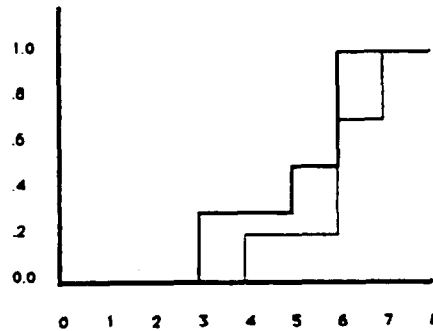

Figure 5: A Comparison of the Two Lotteries

Stochastic dominance is relevant in evaluating an agent's choices in the extended payoff matrix below. Assume that our agent J is faced with the following interaction:

If J considers his own potential outcomes, given the probability distribution he assumes over K's moves, he will reason that he has a .5 chance of receiving the value from either column, given any choice of his moves. Thus, if he chooses move a, he faces a .5 chance of getting either 1 or 3; if he chooses move b, he faces a .5 chance of getting either 2 or 4. Although there is no ordinal dominance here, there is stochastic dominance between the two moves, seen as two separate lotteries (Figure 6).

Thus, a player who was evaluating stochastic dominance would realize that move a was dominated.
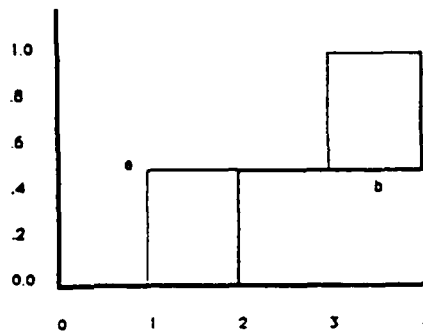
Figure 6: Stochastic Dominance Between Two Moves' Outcomes

3.2.2 Formal Notation for Stochastic Dominance

Since there may be several outcomes with the same payoff to an agent, and since in the probability analysis that the agent performs these outcomes are identical, we would like to "collapse" these identical outcomes in our notation (e.g., combine *all* the chances of getting 3 into a single probability). Thus, we write agent i's subjective probability of getting a certain payoff value, given his choice of move m_i and all his knowledge of the world, as follows:

$$prob_i(v \mid m_i, \xi) = \sum_{\{m_{\bar{\imath}} \mid p(i, \bar{m}) = v\}} prob_i(m_{\bar{\imath}} \mid m_i, \xi).$$

We describe this as the payoff lottery for i given move m_i. When we have

$$\forall x (\int_{-\infty}^{x} prob_i(v \mid c_i, \xi) \, dv < \int_{-\infty}^{x} prob_i(v \mid d_i, \xi) \, dv)$$

we will say that the payoff lottery for i of move d_i is stochastically dominated by the payoff lottery for i of move c_i.[7]

3.3 Utility Dominance

As opposed to ordinal or stochastic dominance, *utility dominance* employs the aggregate measure of "expected utility" which introduces a total order

---

[7]For the reader being newly introduced to stochastic dominance, it might seem odd that c_i's lottery, being everywhere *less than* d_i's lottery, dominates d_i. The curves for better lotteries rise further to the right, and therefore their integrals are smaller up to any given point.

(with equality) over payoff lotteries. The utility function encodes the agent's attitudes toward risky or uncertain payoffs. A utility function which is linear in payoffs will result in expected value decision making.

When the following situation holds,

$$\int_{-\infty}^{\infty} U(v)prob_i(v \mid d_i, \xi)\, dv < \int_{-\infty}^{\infty} U(v)prob_i(v \mid c_i, \xi)\, dv. \qquad (3)$$

then we say that the expected utility for agent i of move c_i dominates the expected utility for agent i of move d_i. Note that the dominating move is on the larger side of the inequality, in contrast to the definition of stochastic dominance, where the dominating move is on the smaller side of the inequality.

4 Rational Moves—A Prescription for an Agent

In this section we describe a prescriptive theory for rational agents in interactions. The criteria for optimality is maximization of expected utility: the agent should choose that course of action which maximizes its expected utility. However, we propose a method which makes use of alternative means of screening moves by which an agent can reduce the number of, and data requirements for, expected utility calculations.

4.1 Rationality Using Ordinal Dominance

We will denote by $R_o(p, i)$ the ordinally rational moves for the agent i in the game p. An individual agent i is said to be exhibiting *ordinal rationality* if it makes moves solely from the set $R_o(p, i)$. The following axiom defines a criterion for eliminating a move from $R_o(p, i)$:

$$pay(i, d_i) <_o pay(i, c_i) \Rightarrow d_i \notin R_o(p, i). \qquad (4)$$

In other words, if d_i is ordinally dominated by c_i (every possible payoff to i of making move d_i is less than every possible payoff to i of making move c_i), then d_i is ordinally irrational for i. Note that this does not imply that c_i *is* ordinally rational, since there may be still better moves available.

4.2 Rationality Using Stochastic Dominance

An individual agent i is said to be exhibiting *stochastic rationality* if it makes moves solely from the set $R_s(p,i)$. The following axiom defines a criterion for eliminating a move from $R_s(p,i)$:

$$\forall x(\int_{-\infty}^{x} prob_i(v \mid c_i, \xi)\, dv < \int_{\infty}^{x} prob_i(v \mid d_i, \xi)\, dv) \Rightarrow d_i \notin R_s(p.i). \quad (5)$$

Thus, if d_i is stochastically dominated by any c_i, then d_i is stochastically irrational for agent i. Note again that this does not imply that c_i is stochastically rational—Equation 5 is a rule to exclude moves from R_s, not to prove that they are members.

4.3 Rationality Using Utility Dominance

An agent is utility rational if it seeks to maximize *expected utility*, as defined in Equation 2. The following axiom defines a criterion for eliminating a move from $R_u(p,i)$, the set of moves with maximal expected utility:

$$\int_{-\infty}^{\infty} U(v)prob_i(v \mid d_i, \xi)\, dv < \int_{-\infty}^{\infty} U(v)prob_i(v \mid c_i, \xi)\, dv \Rightarrow d_i \notin R_u(p,i).$$
$$(6)$$

Thus, if the expected utility of d_i is dominated by the expected utility of any c_i, then d_i is utility irrational for agent i. Note once again that this does not imply that c_i is utility rational. However, this definition of rationality differs from the previous ones in that we know there is a unique member of R_u, or a set of equivalent members (i.e., with the same expected utility). Thus this definition can actually be used to narrow the agent's choices to a single move, given the necessary computational resources to find it.

4.4 The Relationship Among Rationalities

The three definitions of rationality are related in the following ways:

$$d_i \notin R_o(p,i) \Rightarrow d_i \notin R_s(p,i) \wedge d_i \notin R_u(p,i) \quad (7)$$

$$d_i \notin R_s(p,i) \Rightarrow d_i \notin R_u(p,i) \quad (8)$$

$$R_u(p,i) \subseteq R_s(p,i) \subseteq R_o(p,i) \quad (9)$$

The fact that ordinal dominance between two moves implies stochastic dominance between the same two moves for any probability distribution is a simple consequence of their definitions. The fact that stochastic dominance implies utility dominance for any monotonic utility function is a well-known result from decision theory. Stochastic dominance is a robust measure of desirability for the agent, since moves can be eliminated no matter what the risk attitude of the agent as encoded in a utility function.

4.5 Using the Dominance Relations

We will exploit the hierarchy of rationalities (as defined in Equation 9)in our automated agent's activity. Ultimately, he would like to identify the set $R_u(p,i)$, but rather than directly trying to find the utility maximizing move, he can prune his search space by eliminating moves from R_o and R_s. Our agent therefore uses the three-level hierarchy of dominance relations, and their related rationality axioms, as follows:

1. Remove ordinally dominated moves. If a single move remains, select it and finish.

2. Assign and/or determine some properties of $prob_i(m_{\bar{i}} \mid m_i, \xi)$, the probabilities of opponents' moves given each of the agent's possible moves. We admit partial information regarding probabilities because this partial information may be sufficient to eliminate irrational moves in steps 3 and 5.

3. Remove stochastically dominated moves. If a single move remains, select it and finish.

4. Assess and apply a utility transformation to the lotteries defined by the remaining moves.

5. Remove utility dominated moves. All remaining moves will have identical expected utilities. Select one and finish.

Using this technique, our agent is able to maintain his commitment to being a utility maximizer, and still reduce the computational burden of computing expected utility. Information regarding the probability distributions

of opponents' moves is used effectively. The search space can in many instances be radically pruned using this technique.

4.6 An Example

Consider an agent who is confronted with an encounter represented by the payoff matrix in Figure 7.

<div align="center">

K

	e	f	g	h
a	$5\ ^1$	$6\ ^2$	$5\ ^1$	$7\ ^2$
b	$4\ ^2$	$5\ ^1$	$6\ ^2$	$7\ ^1$
c	$4\ ^1$	$3\ ^2$	$4\ ^1$	$3\ ^2$
d	$0\ ^2$	$1\ ^1$	$2\ ^2$	$3\ ^1$

J

</div>

Figure 7: Using Ordinal and Stochastic Dominance

Using ordinal dominance, he is immediately able to rule out moves c and d, leaving him with options a and b. While neither of these moves is ordinally dominated, he would still like to choose between them. He assesses the likelihood that his opponent will make any particular move as equiprobable (perhaps his opponent is only able to reason about ordinal dominance, and thus has no dominated moves; see below, Section 5.2.1). He is then able to conclude that move b is stochastically dominated by move a; move a is thus chosen. Had there been no stochastic dominance, he would have proceeded to compute the expected utility of moves a and b.

In general, however, the probability distributions over opponents' moves will not be readily available, and the computational burden of calculating these probabilities will overshadow the burden of calculating the best move *given* those probabilities. In the next section we describe various approaches where the axiomatic description of opponents allows the agent to deduce

properties of the probability distribution for use in the framework described above.

5 Axioms of Rationality—Description

As described above, the second element of our prescriptive/descriptive approach is a descriptive theory of other agents. We will call the agent to whom we are endowing the prescriptive theory the "agent," and the other agents that we are describing as the "opponents." We describe a framework of rationality that allows us to express many levels of rationality that might be operating in opponents. The ultimate purpose of these axioms is to allow the agent to deduce $prob_i(m_{\bar{i}} \mid m_i, \xi)$ from more fundamental information.

In the remainder of this section, we describe various classes of rationality that this approach can address, and demonstrate how our three-tiered dominance analysis operates in each situation. Finally, we describe how to incorporate uncertainty about what axioms are present in other agents.

5.1 Minimal rationality

An assumption of minimal rationality corresponds to a situation where the agent has no information regarding the rationality of his opponents. This may include a recognition that other players are engaging in potentially irrational behavior (e.g., that the choices the opponents make are independent of their payoffs).

In this case we have

$$\{m_{\bar{i}} : prob_i(m_{\bar{i}} \mid m_i, \xi) > 0\} = M_{\bar{i}},$$

that is, any combined set of moves by the other agents is possible. One version of minimal rationality implies a commitment to equiprobable moves by the opponents:

$$prob_i(m_{\bar{i}} \mid \xi) = prob_i(m'_{\bar{i}} \mid \xi)$$

for all opponents' moves $m_{\bar{i}}$ and $m'_{\bar{i}}$. The effect of this is for the agent to assume that the others will be choosing their moves arbitrarily and ignoring any variation in payoffs.

Minimal rationality does not imply equiprobable assessments. Other information regarding tendencies and biases that opponents have displayed in the past can form the basis for assigning probabilities. The important point is that the assessment is not based on any explicit model of rationality of opponents. It therefore most closely resembles standard decision making under uncertainty, where uncertainty arises from lack of information and randomness in the environment.

5.2 Separate rationality

In separate rationality the agent explicitly admits the possibility that each opponent is rational (to a greater or lesser degree) and has specific capabilities for reasoning about the moves others, including the agent, will take. Below, we examine several types of rationality that might conceivably be exhibited by opponents.

5.2.1 Ordinally Rational Opponents

If the agent assumes that his opponents are at most ordinally rational, then a successive winnowing process can be used to reduce the payoff matrix to a relevant set (this assumes, as well, that the opponents have knowledge of the agent's ordinal rationality; see [18]). Ordinally dominated moves, for both the agent and opponents, are repeatedly removed. The agent then restricts attention to a *reduced* matrix consisting of all ordinally undominated moves (along with opponents' responses). If the remaining set is a single entry then there is a unique solution.

If there are multiple entries, then the opponents and the agent are left with an ambiguity—any of the moves not ordinally dominated are equally desirable. The agent can assume that the opponents will choose arbitrarily within the set of remaining moves—considering the opponents minimally rational as in Section 5.1. The agent is permitted to make this inference because the opponents are only capable of reasoning about ordinal dominance; thus further reasoning about the agent by the opponents is impossible (this was the situation exhibited in the example of Section 4.6).

There is a potential subtlety in using the above method for ordinal dominance. Consider a situation where our agent has several ordinally dominated

moves; does it matter which is "removed" first from the payoff matrix? As it turns out, the order of removal, both for the agent and his opponents. is irrelevant for ordinal dominance; the proof is in [19].

5.2.2 Stochastically and Utility Rational Opponents

Here we address the issue of opponents who, like the agent. are capable of engaging in probabilistic reasoning. Since both agent and opponents can reason probabilistically about each other, there is the potential for infinite regress: the agent's choice is dependent on what he believes his opponents will do, which depends on the opponents' beliefs about the agent. and so on.

One weak form of rationality which lends itself to probabilistic reasoning is due to Strait [21]: if the agent prefers one payoff to another, then his opponent will assign a higher probability to the move with that payoff, and similarly for the agent's assessments of the opponents. One consequence of this principle is that the probability distribution over opponents' moves is dependent on the agent's move, i.e., the agent must consider $prob_i(m_{\bar\imath} \mid m_i, \xi)$. This dependence is not due to a causal linkage, since we are assuming simultaneous action, but rather results from the agent reasoning about the possibility of the opponents "outguessing" him given a particular move.

The foregoing principle results in a set of constraints on probabilities, given our assumption that the payoffs in the encounter are common knowledge and that all players have monotonic utility functions There are various methods for dealing with constraints and/or bounds on probability in decision-making situations.

We can strengthen Strait's principle by adding an assumption that the opponents are Bayesian decision makers. We will restrict our attention to the case where there is a single opponent who is capable of screening moves based on both stochastic and utility dominance relationships. Furthermore, the opponents will be assumed to know that the agent is similarly an expected utility maximizer in making choices.[8]

The infinite regress of reasoning alluded to above is a real concern under these assumptions. One way of dealing with the regress is by explicitly modeling (by way of a probability distribution) the number of levels of regress that the agent believes an opponent will reason, and encoding the agent's

---

[8]This situation more closely resembles the jointly prescriptive theories of game theory.

perception of the opponent's uncertainty at each level. For example, the agent could reason that there is a fifty percent chance that the opponent will reason one level deep, a thirty percent chance two levels deep, a twenty percent chance three levels deep, and zero for all others. This is computationally complex, but is likely to be effective in a world inhabited by computationally limited reasoners.

5.3 Unique rationality

Under unique rationality, the agent assumes that the opponents' moves are fixed in advance, i.e.,

$$prob_i(m_{\bar{\imath}} \mid c, \xi) = prob_i(m_{\bar{\imath}} \mid d, \xi)$$

for all moves c and d to be made by agent i. This can also be expressed as the independence relation,

$$prob_i(m_{\bar{\imath}} \mid m_i, \xi) = prob_i(m_{\bar{\imath}} \mid \xi)$$

which states that the agent's probability distribution does not depend on the move the agent makes. Conceptually, the opponents are assumed to have sealed away their moves before the agent makes his choice. This is orthogonal to the question of *how* the opponents will make their choices; thus, unique rationality can be combined with the various forms of separate rationality presented above, or with minimal rationality. The crucial question here is not whether the opponents are *reasoning* about the agent, but whether their moves will actually be dependent on the move made by the agent (as they are in *informed rationality* below). In certain situations, the assumption of unique rationality allows a technique called case analysis to be applied when computing ordinal dominance (see [17]).

5.4 Informed rationality

Under informed rationality, the agent assumes that opponents have perfect information—they know precisely what move the agent is to take. Informed rationality eliminates uncertainty in the encounter when payoffs are common knowledge. The agent's task in this case is to make a choice that maximizes

his benefit, given that his opponents will respond omnisciently to his move. This is the situation, for example, when there is a time-lag in the making of choices, and the opponents will be able to actually *respond* to our agent's move.

5.5 Uncertainty about Rationalities

In this section we have sketched various classes of rational opponent which our prescriptively designed agent might encounter, and presented some analysis of how each case is analyzed. In general, though, an agent may be uncertain about what class of opponent he faces in a given encounter Probability theory provides a solution—assign a probability distribution to the types of agent which might be encountered, and form a composite distribution over the opponents' moves based on analysis of each case.

6 The Jointly Prescriptive Issues

Ideally, a set of agents who all use the three-level hierarchy of ordinal, stochastic and utility rationality, with coherent probability distributions, will arrive at stable solutions. In general, however, this cannot be guaranteed. Aumann [2] has shown that a construct termed *correlated equilibria* is the result of interactions between utility-maximizing agents. The equilibrium is a probabilistic notion, a generalization of the mixed randomized strategies developed by game theorists. Each agent selects a definite alternative—the uncertainty in the equilibrium is due to the joint uncertainty of the agents about other agents' moves. The existence of correlated equilibria is based on the existence of a common knowledge prior probability distribution over some underlying state of nature. Differences in probability distributions by the agents are the result of differences in information. Though Aumann has provided a characterization of equilibria, it is inherently uncertain due to the uncertainty of the participants and may in fact admit a wide range of possible solutions. Recently Nau and McCardle [16] have shown that correlated equilibria are consistent with a notion of joint coherency in non-cooperative games. This work, however, has not provided an operational procedure for deriving the equilibria based on a single agent's information.

7 Conclusion

The design of automated agents can benefit from the theoretical underpinnings of decision analysis and game theory. Builders of autonomous agents will want to know that their creations are capable of adaptive behavior in the face of various opponents, and can use the "prescriptive/descriptive" aspects of decision analysis to guide their agents' design. The builders of full multi-agent systems will want to ensure certain desirable global properties, and can use the "jointly prescriptive" aspects of game theory to choose the agents' built-in strategies.

We have presented a technique that exploits the relationship among ordinal, stochastic, and utility dominance. Combining it with logical axioms that describe opponents, it is particularly suitable for a deductive engine to use in deciding on a move in an interaction. The technique is based on computational considerations, pruning certain moves before performing computationally expensive operations (such as finding expected utility). We have also presented a sampling of rationality axioms that might be useful to an agent's designer, and given some ramifications of their use. This is basically a prescriptive analysis, discussing one way in which an interacting intelligent agent could be built.

References

[1] Douglas E. Appelt. *Planning natural language utterances to satisfy multiple goals.* Tech Note 259, SRI International, Menlo Park, California, 1982.

[2] Robert J. Aumann. Correlated equilibrium as an expression of Bayesian rationality. *Econometrica*, 55(1):1-18, January, 1987.

[3] John S. Breese. *Knowledge Representation and Inference in Intelligent Decision Systems.* PhD thesis, Stanford University, 1987. Also published as Research Report 2, Rockwell International Science Center, Palo Alto Lab, Palo Alto, California, April 1987.

[4] Michael R. Genesereth, Matthew L. Ginsberg, and Jeffrey S. Rosenschein. *Solving the Prisoner's Dilemma.* Report No. STAN-CS-84-1032,

Computer Science Department, Stanford University, November 1984. Also published as HPP-84-41, Heuristic Programming Project, Computer Science Department, Stanford University, November 1984.

[5] Michael R. Genesereth, Matthew L. Ginsberg, and Jeffrey S. Rosenschein. Cooperation without communication. In *Proceedings of The National Conference on Artificial Intelligence*, pages 51–57, The American Association for Artificial Intelligence. Philadelphia, Pennsylvania, August 1986.

[6] Matthew L. Ginsberg. Decision procedures. In Michael N. Huhns, editor, *Distributed Artificial Intelligence*, pages 3–28, Morgan Kaufmann Publishers, Inc., Los Altos, California, 1987.

[7] Joseph Y. Halpern and Yoram Moses. *Knowledge and Common Knowledge in a Distributed Environment*. Research Report IBM RJ 4421, IBM Research Laboratory, San Jose, California, October 1984. Also published in *Proceedings of the Third Annual ACM Conference on Principles of Distributed Computing*, Vancouver, British Columbia, Canada, 1984.

[8] Eric J. Horvitz, John S. Breese, and Max Henrion. Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning*, 1988. In press.

[9] Joseph B. Kadane and Patrick D. Larkey. Subjective probability and the theory of games. *Management Science*, 28(2):113-120, February, 1982.

[10] Joseph B. Kadane and Patrick D. Larkey. The confusion of is and ought in game theoretic contexts. *Management Science*, 29(12):1365-1379, December, 1983.

[11] R. Duncan Luce and Howard Raiffa. *Games and Decisions, Introduction and Critical Survey*. John Wiley and Sons, New York, 1957.

[12] Kurt Konolige. *A first-order formalization of knowledge and action for a multi-agent planning system*. Tech Note 232, SRI International, Menlo Park, California, December 1980.

[13] Kurt Konolige. *A Deduction Model of Belief and its Logics.* PhD thesis. Stanford University, 1984.

[14] Robert C. Moore. *Reasoning about knowledge and action.* Tech Note 191, SRI International, Menlo Park, California, 1980.

[15] Robert C. Moore. *A formal theory of knowledge and action.* Tech Note 320, SRI International, Menlo Park, California. 1984. Also in *Formal Theories of the Commonsense World,* Hobbs. J.R.. and Moore. R.C. (Eds.), Ablex Publishing Co. (1985).

[16] Robert F. Nau and Kevin F. McCardle. *Coherent Behavior in Noncooperative Games.* Working Paper 8701, The Fuqua School of Business, Duke University, Durham, North Carolina, 1988.

[17] Jeffrey S. Rosenschein. *Rational Interaction: Cooperation Among Intelligent Agents.* PhD thesis, Stanford University, 1986. Also published as STAN-CS-85-1081 (KSL-85-40), Department of Computer Science. Stanford University, October 1985.

[18] Jeffrey S. Rosenschein. The role of knowledge in logic-based rational interaction. In *Proceedings of The IEEE Phoenix Conference on Computers and Communication,* IEEE, Scottsdale, Arizona, March 1988. pp. 497–504.

[19] Jeffrey S. Rosenschein and John S. Breese. *Communication-Free Interactions Among Rational Agents: A Probabilistic Approach.* Technical Report, Rockwell International Science Center, Palo Alto Laboratory. 1988. Forthcoming.

[20] Jeffrey S. Rosenschein. *Interaction Representation: Focal Points and a Call for New Ways of Representing Encounters.* Technical Report, Rockwell International Science Center, Palo Alto Laboratory, 1988. Forthcoming.

[21] R. Scott Strait. *Decision Analysis of Strategic Interaction.* PhD thesis, Stanford University, 1987. Also published as UCRL-53825, Lawrence Livermore National Laboratory, Livermore, California, December, 1987.

[22] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior.* Princeton University Press, Princeton, 1947.

[23] John Wilson. Subjective probability and the prisoner's dilemma. *Management Science*, 22(1):45-55, January, 1986.

[24] G. A. Whitmore and M. C. Findlay, editors. *Stochastic Dominance: An Approach to Decision Making Under Risk.* D. C. Heath and Company. Lexington, Massachusetts, 1978.

ARTIFICIAL INTELLIGENCE AND DESIGN: OPPORTUNITIES, CHALLENGES, RESEARCH PROBLEMS AND DIRECTIONS*

Saul Amarel
Department of Computer Science
Rutgers University

July 1988

ARTIFICIAL INTELLIGENCE AND DESIGN: OPPORTUNITIES, CHALLENGES, RESEARCH PROBLEMS AND DIRECTIONS

Saul Amarel
Department of Computer Science
Rutgers University

Abstract

The issues of industrial productivity and economic competitiveness are of major significance in the US at present. By advancing the science of design, and by creating a broad computer-based methodology for automating the design of artifacts and of industrial processes, we can attain dramatic improvements in productivity. It is our thesis that developments in computer science, especially in Artificial Intelligence (AI) and in related areas of advanced computing, provide us with a unique opportunity to push beyond the present level of computer aided automation technology and to attain substantial advances in the understanding and mechanization of design processes. To attain these goals, we need to build on top of the present state of AI, and to accelerate research and development in areas that are especially relevant to design problems of realistic complexity. We propose an approach to the special challenges in this area, which combines 'core work' in AI with the development of systems for handling significant design tasks.

We discuss the general nature of design problems, the scientific issues involved in studying them with the help of AI approaches, and the methodological/technical issues that one must face in developing AI systems for handling advanced design tasks. Looking at basic work in AI from the perspective of design automation, we identify a number of research problems that need special attention. These include finding solution methods for handling multiple interacting goals, formation problems, problem decompositions, and redesign problems; choosing representations for design problems with emphasis on the concept of a design record; and developing approaches for the acquisition and structuring of domain knowledge with emphasis on finding useful approximations to domain theories. Progress in handling these research problems will have major impact both on our understanding of design processes and their automation, and also on several fundamental questions that are of intrinsic concern to AI.

We present examples of current AI work on specific design tasks, and discuss new directions of research, both as extensions of current work and in the context of new design tasks where domain knowledge is either intractable or incomplete. The domains discussed include Digital Circuit Design, Mechanical Design of Rotational Transmissions, Design of Computer Architectures, Marine Design, Aircraft Design, and Design of Chemical Processes and Materials. Work in these domains is significant on technical grounds, and it is also important for economic and policy reasons.

ARTIFICIAL INTELLIGENCE AND DESIGN: OPPORTUNITIES, CHALLENGES, RESEARCH PROBLEMS AND DIRECTIONS

Saul Amarel
Department of Computer Science
Rutgers University

I. INTRODUCTION: BACKGROUND

The issues of industrial productivity and economic competitiveness are taking center stage in the US at present. There are many factors that impact on these issues. Some are social, economic and political, and some are technological. We will focus here on a set of technological and scientific developments that promise to have a strong impact on productivity improvement. In particular, we will concentrate on the role that Artificial Intelligence (AI) and related areas of advanced computing can play in this important area.

a. The challenge of productivity improvement

To attain major improvements in industrial productivity, the following capabilities are critical:

> rapid reaction to changes in functional requirements of products and to new technological opportunities;

> rapid transition from design concept to product;

> production of high quality products at the lowest possible life-cycle cost. Costs should take into consideration design and manufacturing efforts as well as testing activities, operation and maintenance.

More specifically, the reduction of **design time**, and the reduction of **time to plan an efficient manufacturing process** , are critical for productivity improvement. Naturally, it is also important to reduce the time needed to setup a manufacturing process, and the time required for producing (actual manufacturing and testing) a high quality product.

In a recent *Science* article, John Young president and CEO of Hewlett-Packard, and Chair of the President's Commission on Industrial Competitiveness, comments [Young 1988] :

> "In today's world, shortening the time between idea stage and finished product often makes the difference between success and failure. The high costs of developing new products, the brief time before copies appear, and rapid obsolescence make for a short innovation cycle - often 3 to 5 years [Press 1987]."

In discussing Hewlett-Packard's experience with efforts to change basic approaches to design and to bring closer the design and manufacturing functions, he states:

"... fully 25% of our manufacturing costs were involved in responding to quality problems - that is, not doing things right the first time."

Consistent with these comments, we can summarize as follows key goals for productivity improvement:

(i) **do it right rapidly**, and

(ii) **do it right the first time** - by avoiding adjustments that show up later during the manufacturing and testing stages, and that are not foreseen in the initial design process.

These goals impose strong requirements on the design process, and on the integration of design and manufacturing.

b. Computer aided approaches to design and manufacturing

(i) Current State

There has been considerable progress in recent years in the development of computer-aided design and computer-aided manufacturing (CAD/CAM) systems and their application to various industries. In particular, such systems are being used widely in the electronics, computer and machining industries [see 'Toward a New Era in US Manufacturing' 1986].

Most of the work in the CAD/CAM area has concentrated on relatively **low levels of design and manufacturing** tasks. Typically, current systems include tools for: **representing and editing** a piece of a finished design; **evaluating and analyzing** proposed designs; **describing and simulating** manufacturing processes; and **monitoring and controlling** manufacturing processes - mostly, in open loop mode.

In general, the degree of **integration** between CAD and CAM systems has been relatively modest in the past.

Now, the description of a finished design, or of a well-defined manufacturing process, are each a **solution** to some high level design or manufacturing problem. Thus, most of today's CAD and CAM tools are concerned with the representation, manipulation and testing of solutions to design and manufacturing problems - not with the handling of the problems themselves and with the efforts to solve them.

In general, computer techniques have been used very little to provide intellectual assistance in **early stages of design** (what is usually called conceptual design), and to keep track of design options, of incomplete design ideas, of the reasoning behind various design decisions, and of the general evolution of the design process.

Also very little has been done about retaining **design experience** in computers, so that it can be used to improve future designs.

(ii) New technical opportunities: a thesis

Recent developments in computer science - especially in Artificial Intelligence, and also in information systems technology, and in large scale modelling and simulation - provide us with **unique opportunities** to push beyond the present level of computer aided automation technology and to attain fundamental improvements in industrial productivity.

Our vision of the next generation of automation technology includes computer systems for **high level design processes**, where a product is designed for functionality, manufacturability, maintainability and economy. Also, these systems would facilitate the **integration** of design and manufacturing functions. Under these conditions, product quality would be improved, and the time interval between initial design concept and product would be shortened appreciably. Systems with these capabilities can be realized by bringing to bear new advances in Computer Science, and in particular in AI, to the automation of design processes.

We are also assuming that a powerful **infrastructure for design and manufacturing** can be built on basis of ideas and techniques that are being developed in AI and in Information Systems. Such an infrastructure would include **design knowledge bases** for products and processes that could be widely accessible by researchers and engineers.

In short, it is our _thesis_ that the computer field is now at a point where it can provide the intellectual foundations and the technical basis for developing a science and technology of design and manufacturing that will have a dramatic impact on industrial productivity.

By building on top of the present state of computing, and by further accelerating research and development in areas of advanced computing that can contribute to substantial improvements in design and manufacturing, we can bring about major gains in national productivity and competitiveness.

c. A national initiative in Computer aided productivity (CAP)

Based on this thesis, a preliminary version of a plan was developed within DARPA in 1987 which proposed the launching of a major new **national initiative,** involving government, universities and private industry, that would focus on:

> (i) **research and development** in several areas of computing that promise to have strong impact on automation of advanced design and manufacturing processes.

> (ii) **development, construction and operation of experimental testbeds** (system demonstration efforts) of sufficient size and scope to try new ideas about computer aided design in realistic settings, and to aid in technology transfer.

> (iii) **dissemination of ideas and training** in the new computer aided methods and technologies.

51

Emphasis would be placed on Univ rsity-Industry collaboration. The <u>choice of domain</u> for the system demonstration efforts was recognized as an important parameter of the plan. A variety of <u>criteria</u> were used to obtain an initial set of domain options. Some of the criteria were **technical** (e.g., expectations that work in the domain would help identify key research issues in design, and they would lead to the development of computer based solutions for important classes of design problems); others were concerned with the **economic** and **policy significance** of the domain; and still others were concerned with issues of **feasibility** (e.g., current state of ideas, technology and experience in the domain, likelihood of interdisciplinary collaboration). The initial exploratory study of domain possibilities resulted in the following set of options:

> **VLSI design and fabrication.** Considerable amount of DARPA-supported effort already exists in this area.

> **Design of computer architectures;** design and manufacturing of computer assemblies.

> **Design and manufacturing of complex mechanical assemblies,** in particular aircraft and marine structures.

> **Design and production of new materials and chemicals.**

> **Design, production and maintenance of software.** Success in this area provides enormous leverage for the advancement of industrial automation in many areas.

> **Robot design and manufacturing.** Work in this domain forces attention on the design of heterogeneous systems that include mechanical, electrical/electronic and software components; it also provides productivity leverage as it increases the availability of advanced flexible production tools.

The current state of computer science and technology would provide a good starting point for the CAP national initiative. However, it was recognized that extensive research would be needed in several areas of computing in order to advance CAP goals.

More specifically, it was recognized that **basic work is needed in several areas of AI , as well as in large scale computing, in databases, robotics, computer architectures, distributed and networking systems, and interfaces.** The preliminary CAP plan concluded that we need to strengthen and accelerate current research in these areas, and to initiate extensions with the design-manufacturing theme in mind.

Since the middle of 1987, the CAP preliminary plan has undergone various changes, and it has been reviewed in various forums; however, it has not reached yet a final form, and consequently no decisions have been made regarding implementation of key ideas in the proposed initiative.

The research directions and approaches discussed below are in the general spirit of the CAP concept, and they are based on the underlying assumptions that were used in CAP planning. We believe that it is still possible, and valuable, to pursue in some depth **parts of the CAP concept,** even if a full-fledged CAP initiative does not materialize in the near future.

In this paper, we will concentrate on those parts of CAP that relate to AI research. In particular, we will focus on AI issues that relate to problems of design. We will then discuss current work in this general area; and we will outline directions of future research that grew out of the preliminary planning for the CAP initiative.

II. GENERAL ISSUES IN ARTIFICIAL INTELLIGENCE AND DESIGN

Many of the key conceptual issues in AI and Design were discussed as early as 1969 by Simon in (the first edition of) his pioneering book 'The Sciences of the Artificial' [Simon 1981]. In that book Simon argues that a **science of design** is not only possible but it is already emerging within the general framework provided by AI.

Let us consider the nature of design problems. From the point of view of AI, the central challenge is how to formulate, represent and solve a broad range of design problems; in particular, how to develop systems that will generate automatically or quasi-automatically (in interaction with people), one or more designs for an artifact (a product) or process in response to given specifications and constraints, i.e., to desired design goals.

Typically, the problem specifications include requirements on functional properties of the design. The constraints may include conditions on the structure of acceptable designs (e.g., limits on total complexity of the design), assumptions about desirable modes of manufacturing, testing, and maintenance of the proposed designs, and restrictions on resources that may be available (time, etc.) to obtain a design. The design specifications and constraints may be incomplete at the beginning of the design process; in general, they will be changing in the course of the process - often because of feedback from attempts to solve the design problem. In many real-life design problems, the specifications and constraints impose a **multiplicity of goals** on the design process. In these cases we face the issue of "concurrent design", i.e., how to control decisions during the problem solving process so that the multiple goals are all taken into account in the effort to generate a candidate design. For example, in digital circuit design, this involves guiding the initial design not only on basis of the desired functional (input-output) characteristics, but also taking into consideration requirements about physical layout on a chip and testability.

The formulation of a design problem requires a specification of the language of design structures (e.g., components, rules of aggregation of components) in terms of which a design is to be described. It is the language in which solutions to the design problem are expressed. Such a language should provide means for describing designs at **various levels of completeness and resolution.** Structural descriptions of designs should facilitate processes of incremental solution construction and modification, as well as the use of solutions in redesign, analogical design, explanation and training.

In many cases of interest, the language of design structures is different from the language in which design specifications and/or constraints are expressed. More importantly, the set of concepts and abstractions used to describe design goals on the one hand and design structures on the other may be quite different. This situation leads to design processes where **reasoning takes place in two spaces - the space of specifications/constraints and the space of structures.** An important issue in this area is how to coordinate effectively the processes of reasoning in the two spaces.

A key component in the formulation of a design problem is the specification of the domain in which the problem is embedded. A domain specification is a body of concepts (a set of theories and models) in terms of which specific problems in the domain can be expressed, understood, and processed. It includes definitions of objects and predicates that enter in the specification of problems in the domain, relationships among them, and special properties of the problem environment under consideration. The specification of design goals, and also of the language of design structures, is in terms of concepts in the domain specification. For example, problems of digital circuit design may be expressed in terms of concepts in a domain that includes circuit theory, switching theory, and a body of knowledge about prototypical digital designs and their properties. Similarly, problems of boat design may be handled in a domain that includes fluid dynamics, theories of structures and materials, relevant approximate theories and models in these areas, and a body of knowledge about relevant classes of boat designs.

The nature of knowledge in a domain specification has a crucial influence on the kind of design processes that are possible in the domain. The completeness and accuracy of relevant knowledge are clearly important. Also, if domain theories are complex and intractable, we face serious issues of computational complexity. In these cases, it is essential to find **appropriate abstractions, models and approximations** to assist in reasoning about structure-function relationships in a problem, and, in general, to render problems computationally tractable.

The process of constructing a solution to a design problem amounts to the generation of (the description of) a design that satisfies the given design specifications and constraints. The solution is expressed in the language of design structures. The nature and efficiency of such a process is strongly influenced by the choice of representation of domain knowledge, of problem goals, and of the language for articulating design structures. It is desirable to represent design structures, in such a way that **structural parts of a design can correspond as directly as possible to parts of design specifications. In other words, structure should have meaning in terms of function.** Unfortunately, this condition is hard to attain, except in design domains that are highly developed and very well understood. In many cases, the key conceptual difficulty in solving design problems can be traced to difficulties in satisfying this condition.

The problem solving approach to design is strongly influenced by the way in which design specifications and constraints can be used to control solution construction. Specifications and constraints can be used in two major ways:

   (1) by *a posteriori* testing whether a candidate design satisfies them; and

   (2) by *a priori* constraining the generation of possible design structures to be consistent with them

The *a posteriori* use of design goals involves analysis and evaluation of a candidate solution, and an assesment of the degree to which the candidate satisfies these goals. Often, such an evaluation involves simulation, or, more generally, the computation of functional characteristics of a candidate structure that could be directly compared with the goals of the design problem, so that a decision can be made whether the candidate is an acceptable solution to the problem. Typically, **domain knowledge is available in a form that can be used easily for an effective evaluation of candidate design structures.** In many cases of interest, evaluation is possible only if candidate solutions are completely specified, i.e., design structures at intermediate stages of

construction cannot be evaluated. In these cases, approaches to solution construction are limited to variants of weak generation-and-test methods, possiby supported by hill climbing in the space of design structures.

In complex domains, we also face situations where the problem of computing the functional properties of a design structure is very demanding in terms of computational resources. Issues of finding and using appropriate approximate or specialized theories are extremely important in these cases. Examples of design tasks where a *posteriori* use of specifications are quite common, and where computational requirements are very heavy, are boat design and aircraft design. In these cases, computational evaluation of a candidate design involves the use of fluid dynamics theory to compute key dynamic properties of the design, such as drag.

The a *priori* use of design goals involves analysis and transformation of the design specifications and constraints, so as to enable them to control directly the generation of solutions to the design problem. An important approach is to **decompose design goals**, and to handle each of the resulting design subproblems separately. Here we encouter the important issue of problem decomposition and the related issues of how to handle dependencies between subproblems. More specifically, the problem is how to find decompositions that are logically sound ,i.e., they lead to a valid solution, and are also computationally efficient, i.e., they reduce computational complexity by breaking a problem into independent (or weakly dependent) subproblems. In order to reason directly from specifications to design structures, it is important to have available substantial domain knowledge about relationships between functional properties of a design - as described in the specifications - and its structure. Furthermore, **such knowledge should be available in a form that facilitates processes of reasoning from function to structure.** This requires the ability to find, with relative ease, solutions to inverse problems in the domain, i.e., finding a (small) set of design structures that satisfy given functional specifications. For example, in boat design this involves the ability to compute the geometric/structural characteristics of a boat's hull and keel configuration from the specification of desired hydrodynamic properties for the boat. In general, inverse problems are much harder then 'direct problems', where the task is to find functional characteristics of a given structure.

Experience shows that problem solving power depends on the degree to which problem conditions (i.e., the design specifications and constraints in our present case) can be made to influence directly the process of solution construction. The amount of influence exerted by problem conditions depends on their relative use in an a *priori* and an a *posteriori* mode. The more dominant the a *priori* mode, the more powerful the problem solving process. In previous work, we had introduced a classification of problems along a spectrum where problems are ordered by the degree to which their problem conditions can be used to control directly the process of solution generation [see Amarel 1987]. At one end of this spectrum, we have derivation problems, where the mode of control is mainly a *priori;* at the other end, we have formation problems, where the control is mainly a *posteriori.*

Design problems are spread over the derivation-formation spectrum, with a preponderance of problems close to the formation end of the spectrum. Examples of design problems that are mainly of derivation type occur in digital circuit design; examples that are mainly of formation type can be found in boat design. To date, most of the work on design problems in AI has been on derivation type problems. In general, there has been much more work and accumulated experience in AI with derivation problems than with formation problems. Thus, by focusing on design problems that are

close to the formation end of the spectrum, we can expect to advance not only the science and technology of design but also the basic understanding of formation problems in AI.

An important way of increasing system performance in a design task is to use knowledge of the domain, as well as design experience, to shift the representation of the design problem so that it can be seen as moving over the derivation-formation spectrum in the direction of the derivation end. This involves the acquisition and shaping of knowledge in a form that permits increased direct control of the solution generation process by the design specifications. In order to achieve this, it may be necessary to explore and experiment in regions of design space, and to search for patterns that can lead to 'local' theories of structure-function relationships. This is an area which can benefit from current AI work on theory formation and discovery [see Amarel 1983, Amarel 1986]. Conversely, by approaching these issues of shifts of representation in the context of design tasks, we are likely to identify problems and approaches that will advance basic work on discovery and theory formation.

In order to work with realistic designs, it is essential to develop systems that can handle **high levels of complexity** (hundreds of interacting design specifications and constraints, great variety of types of constraints). This requirement presents a new challenge for AI research. It is not clear that currently available methods for handling planning or design problems with a relatively small number of interacting goals can scale up to problems of much higher complexity. This is an area that needs empirical exploration, and (most probably) new ideas. A promising approach is to develop systems that are able to use a small number of generic methods for controlling design processes, and that have effective methods for assimilating domain-specific information to increase the power and efficiency of design in specific domains.

A design system should be able to keep a **record of the design process**, i.e., the reasoning process that establishes a bridge between given specifications and constraints of a desired object on the one hand and a description of the object, in the required 'final' form, on the other. Such a record will be needed for purposes of incremental generation and modification of a solution, for analysis, explanation and training.

Design records should be available in an (appropriately structured) design knowledge base to provide an experiential basis for future designs. Such a knowledge base can be used for processes of redesign (they represent a large portion of practical design activities), for processes of design by analogy, for training of designers, and for processes of forming design theories in specific areas - via distilling information in the knowledge base and shaping it into a form which is especially appropriate for handling certain classes of design tasks with great efficiency.

Progress in AI methods for theory formation and learning is opening the way to the development of systems that can automatically acquire design expertise from design experience. In some situations, theory formation and discovery methods are central elements in systems for the automatic generation of designs An interesting example is the design (discovery) of new materials, and of processes for manufacturing them, on basis of a corpus of experience about other, similar, materials.

Issues of training (of designers, system operators and system maintainers) are of great relevance to issues of productivity; and they should receive special attention during the design process. By introducing the concept of a comprehensive design knowledge base - where a design is represented by a structured entity including specifications,

constraints, representations of the design object, and the trace of reasoning that leads to the design object - we make it possible to handle issues of documentation and training as integral parts of the design task.

Another important issue that should be considered in exploring AI approaches to design is the broader context within which a design problem is formulated. In many cases of interest, the setting of design specifications is itself a problem solving activity which is responsive to higher level goals, or to goals that are ill-defined. In these cases, design specifications can be seen as subgoals that are derived from higher level goals via analysis, refinement and reduction. For example, functional specifications for the design of a digital circuit, may be obtained as a subgoal in the course of solving a higher level problem of computer design. Similarly, the specification of desired hydrodynamic properties of a yacht's hull-keel-winglet assembly, may be derived from a higher level planning problem whose goal is to win a race.

Often, the problem of formulating design specifications from higher level goals must be handled within a system which is theory-limited or/and where relevant knowledge is uncertain and incomplete. Yacht design driven by the goal of winning a race is an example of such a situation. In these cases, the design specifications may have to change several times in a process of 'generate-and-test', which involves cycles of design, implementation, physical testing, and evaluation of the design in light of the desired high level goals. Thus, a design problem can be seen as residing in an inner loop of a larger loop which characterizes the process of producing an object that satisfies the higher level goals. The implications on AI and design are as follows: The real challenge is not how to solve a single, well-formulated, design problem; but **how to organize the solution of a family of related problems whose formulations differ 'relatively little' from each other.** This has interesting implications on the choice of representations and control mechanisms to handle the design task. In particular, issues of redesign and design modification become central.

The formulation of a design problem is influenced not only by the higher level goal environment in which it is embedded, but also by its implementation context, e.g., by the manufacturing and maintenance processes for implementing the design and for supporting its integrity. As indicated previously, some of the constraints that enter in the problem formulation, and the specification of the language for design structures, should reflect assumptions about the implementation context. Often, these assumptions are inaccurate or incomplete, or changes may occur in the implementation environment as time unfolds. Again, these situations may lead to reformulations of the design problem, and consequently to redesign activities. This has also implications on the organization of systems for handling the design task.

Because of these issues of context, **design problem solving should be viewed as involving a multiplicity of related problem solving episodes.** This is quite different from the conventional viewpoint in AI which has concentrated on a single problem solving episode, without much 'transfer' occuring between episodes. Therefore, in studies of AI and design, increased attention should be given to 'longitudinal approaches' that would focus on the behavior of a design system over relatively long periods of time. Such a system would remember previous problem solving episodes, and it would use them appropriately for handling current tasks. The situation in design is similar to situations in planning problems where changes in the state of knowledge about the problem induce a succession of related plan generation and plan modification episodes. Fortunately, the recent emergence of work in case-based reasoning, in reasoning by analogy, and in learning problem solving is producing ideas and systems

that can provide starting points for research on longitudinal approaches to design problem solving.

Another aspect of the dynamics of design problem solving is the acquisition of design expertise with experience, which leads to improved design capabilities. This can be seen as a major developmental process for design systems. In general, the dynamic characteristics of design problem solving - at the performance level and at the developmental level - have important implications on memory organization, and, more specifically, on the building and maintenance of appropriate knowledge bases of designs.

III. DIRECTIONS FOR BASIC AI RESEARCH

In view of the issues on AI and design that we outlined above, and looking at AI research from the perspective of impact on the automation of design, the following are directions of basic AI research that need to be pushed and strengthened. We organize them in three groups: solution methods, representations, and knowledge handling.

(a) Solution Methods

Development of methods for solving design problems with multiple interacting goals of various types; procedures for analysis and for effective handling of complex systems of constraints; approaches to the solution of formation problems where problem goals are used mainly in an *a posteriori* mode for control of solution generation; methods for coordinating the bottom-up generation of candidate solutions with the top-down reasoning about design specifications and constraints; approaches to hierarchical solution of design problems, and related methods of constraint relaxation and approximate optimization.

Development of effective methods for decomposing design problems into loosely coupled subproblems, for handling subproblem interactions, and for combining partial solutions; as an important special case, methods for partitioning goal sets of a design problem so that each partition can be handled nearly independently, and the resulting solutions for each partition can be readily assembled to obtain a global solution to the problem; improved systems for handling reduction, refinement and conflict resolution processes.

Development of methods for reasoning with qualitative and quantitative information; procedures for effective coordination of mathematical models/methods with symbolic reasoning and heuristic search. These are essential for bridging conventional engineering methodologies with AI problem solving methods applied to real-life design tasks.

Development of systems for effectively handling a multiplicity of closely related design problems; approaches to redesign, design from prototypes and design by analogy; methods of organizing in large memories records of previous design cases - to support solution generation for a current problem by adaptation/modification of solutions to stored cases of 'similar' design problems.

(b) **Representations**

Development of representations for design processes at different stages of completion and at different grains of detail; approaches to choosing representations that are 'best suited' for specific design tasks (in terms of computational efficiency), and methods for managing and coordinating multiple representations (views) of a design; approaches to shifting representations in a manner that increases performance in specific design tasks.

Of central importance is the organization of a knowledge base of design records, where each record includes a trace of decisions that shows why a particular solution to a design problem satisfies the specifications and constraints of the problem, and is structured in a way that can be effectively used for such processes as explanation, redesign, and design by analogy. Good progress has been made to date in the development and use of design records for problems of derivation type, where each part of a solution structure has a 'raison d'etre' in terms of specific design specifications/constraints, e.g., in certain problems of digital circuit design [see VEXED, in IV.a.(i) below]. The situation becomes more difficult in problems of formation type where it is rare to have a justification for each part of a solution structure in terms of specific design goals. In problems of this type, a given functional property of the design, or the satisfaction of a global constraint by the design, can only be traced to the combined action of large subassemblies of the design structure, possibly to the entire design structure. Many problems of boat design or aircraft design are of this type. More basic work is needed on the construction and use of design records for design problems that are close to the formation end of the derivation-formation spectrum.

(c) **Knowledge Handling**

Development of approaches for the acquisition and shaping of domain knowledge in ways that lead to major improvements in scope and efficiency of design systems; methods for finding useful approximations to domain theories, preferably in parts of the domain that are especially relevant to problems of interest; in particular, emphasis on invertible approximations that permit reasoning to proceed from function to structure; formation and use of models and specialized theories for parts of the design domain, and finding qualitative properties of theories that are useful for guiding specific design tasks.

The evolution of a design domain, and the improvement of design expertise in the domain, are marked by the development of more accurate domain theories and computationally efficient approximations of them, the reformulation of knowledge from a form that is mainly oriented to an *a posteriori* evaluation of candidate solutions to an *a priori* control of solution construction, and the creation of subdomains with specialized representations and methods. The study of these knowledge transitions will lay the groundwork for computer-based tools to support them.

Current work on machine learning and automatic acquisition of problem solving expertise needs to be pursued further in the context of design problems. Of particular interest is extension of work on design apprentices [see LEAP, in IV.a.(iv) below]. Continuation of basic work on theory formation and machine

discovery is needed to support research on the knowledge transitions that characterize the developmental aspects of design systems.

Research in these areas is expected to have major impact both on our understanding of design processes and their automation, and also on several fundamental questions that are of intrinsic concern to AI as a science. Thus by focusing on the challenge of automating design processes, we are also providing an effective vehicle for pushing research in several basic areas of AI.

It is essential that basic work in AI along the directions just discussed be carried in parallel with (or as part of an attempt to handle) specific design tasks. One or the obvious advantages of this is that it provides a mechanism for effective testing of ideas, as well as for assessing their limitations and identifying new problems. Another important (but less obvious) advantage is that it induces joint consideration of (i) solution methods, (ii) representations and (iii) knowledge handling in the context of a single task. The task plays the role of an integrating agent, and it facilitates the understanding of interactions between control, representations and knowledge.

We will discuss next current efforts in AI and design, as well as directions of future work. Since the design task is an important organizing factor in this area (both in pulling and integrating the research efforts, and also for project management), we will proceed by discussing projects that focus on specific design domains and on specific tasks in these domains. It is not our intention to provide here a complete survey of work in the field. There is a growing number of publications that cover current work in considerable breadth and depth [see Tong 1987a, Tong and Sriram 1988]. We would like, however, to give a sense of the types of current and planned design tasks on which AI work is focusing, and to discuss some of the experience obtained so far.

We will proceed therefore by presenting brief summaries of a few illustrative projects. Our examples are taken mostly from work at Rutgers, or from planned collaborative work which involves investigators at other institutions as well as at Rutgers.

Another note regarding our choice of design domains and tasks. We are focusing attention on domains that were found to be of high priority in the preliminary CAP studies - on basis of technical, policy/economic and feasibility criteria [see section I.c above].

IV. EXAMPLES OF CURRENT WORK ON DESIGN TASKS

a. Digital Circuit Design

(i) VEXED [Mitchell et al 1985a, Steinberg 1987]

Research in the VEXED project started at Rutgers about four years ago. The VEXED system assists in the design of digital circuits from input-output specifications down to the transistor level. It is implemented as part of an interactive circuit editor.

Design specifications are presented as desired functional (logical) and timing properties of circuits. The language of solutions can be used to describe circuit structures at various levels of abstraction - in terms of abstract modules, datapaths, datastreams, and elementary transistor circuits. The key body of knowledge used by the

system is a set of "implementation rules" for reasoning from functional goals to circuit structures. Examples of such implementation rules, paraphrased in English, are:

IF the goal is to convert parallel to serial, THEN use a shift register.

IF the output at time t2 depends on an input at time t1, THEN one way to implement a module is as a memory submodule, which holds the input value from t1 to t2, and a second memory submodule which uses this stored value at t2 to compute the output.

The implementation rules can be used to reason in a top-down mode. Each rule specifies how a circuit module which is to perform a desired function can be specified as an aggregate of submodules, each of which is to perform an appropriate component function. The application of an implementation rule can be seen as a refinement of the design. This is so because an incompletely specified design structure becomes further refined, i.e., more specified, by the rule application. Typically, a refinement represents a decomposition of a design problem into subproblems. However, it is often difficult to completely specify a decomposition via an implementation rule, because of the difficulty of defining explicitly all the interactions between subproblems. On basis of domain knowledge about digital circuits, interactions between subproblems in a refinement step can be specified in the form of constraints at the interfaces between submodules of the refined module. As submodules become further refined, interfaces become further constrained. The VEXED system has a powerful constraint propagation subsystem, called CRITTER [Kelly 1985] that communicates appropriate constraints to parts of a circuit structure that are affected by constraints in other parts. Thus one-or-more steps of refinement + constraint propagation have the effect of completely decomposing the original design problem into a set of 'elementary' design problems whose solution is known. This method of solution construction can be seen as a form of relaxed reduction [Amarel 1987, Amarel 1983].

The trace of reasoning that starts with functional specifications for a circuit, and proceeds via a sequence of refinement + constraint propagation steps to specify the circuit structure, is called the design plan associated with the circuit, and is used as the design record. This can be seen as a structural description of the circuit in terms of aggregates (structural parts, modules) at different levels of abstraction. These structural parts of the solution are defined by the refinement steps that were used by the system in the course of generating the solution. Also, the design plan can be seen as a justification/explanation of why the various parts of the structure were selected to perform their specific local functions in view of the global functional requirement imposed on the circuit.

The design problem handled by VEXED is a derivation problem. The design goals have a strong direct influence on solution construction. In particular, the system can reason from a global functional requirement to local functional requirements of parts of the circuit, and from them to structural definitions of circuit parts and their interconnections.

The domain specification of VEXED includes (i) knowledge about relationships between structure and function in digital circuits, about signal timings and about encodings, and (ii) a taxonomy of component types (e.g., memories, boolean circuits). The implementation rules can be seen as part of the domain specification. Alternatively, they can be seen as part of the specification of the language of solutions, in particular, as the 'grammar rules' that determine how circuits are to be structured.

The mode of interaction with VEXED is as follows: the user selects a design (sub)task; the system suggests possible refinements; the user selects a refinement; the system carries out the refinement, propagates and checks new constraints, maintains a design record, and presents to the user the state of the design process. Thus the user is responsible for control of the design process - both attention control, i.e., on what task to focus next, and for move selection, i.e., what implementation rule to choose.

The emphasis in the VEXED project is on **representational and knowledge structuring issues** in the digital circuits domain, as well as on mechanisms for maintaining consistency between constraints in different parts of a candidate design, and for managing design records.

The present state of VEXED is as follows. The system includes approximately 50 implementation rules, covering most of standard NMOS designs of boolean functions, plus some latches. Students in an introductory VLSI class at Rutgers have used the system to design simple circuits (e.g., full adders). The system is slow, largely because of the **time cost of constraint propagation**. It takes approximately 5 minutes to design a circuit with 20 modules on a Xerox 1109.

Recently, the domain-independent aspects of VEXED, i.e., the part that implements the method of refinement + constraint propagation, were abstracted into a system called EVEXED, and EVEXED was used to reimplement VEXED and also to implement MEET, a system that designs mechanical systems for transmission of rotational power [see IV. b. below].

Research in the project is now focusing on the problem of **scaling up the complexity of designs that VEXED can handle by reducing the time complexity of constraint propagation**. Another problem that will receive increased attention is how to develop a system architecture that embodies a general method of design (in particular, refinement + constraint propagation) together with means of "compiling" information about a specific design domain expressed in some general formalism into an efficient specialized representation and program. This is part of a more general goal of **how to handle more effectively knowledge acquisition and restructuring** in design problems.

Other problems that were identified in the course of work with VEXED is the issue of **automating control decisions** in the course of design; the issue of **multiple interacting goals** in design, in particular accomodating constraints on resources; approaches to **redesign**; and the automatic **learning of design rules**. Several projects that are closely related to VEXED have been concentrating on these issues. We will discuss them briefly in the following.

(ii) DONTE [Tong 1987b, Tong 1988]

Research on the DONTE project started at Stanford and Xerox about four years ago, as a successor to the Palladio project [Brown et al 1983]. Work on the project is continuing at Rutgers. The DONTE system designs digital circuits to meet given functional specifications (as in VEXED) and **resource constraints**. The basic model of design is similar to the top-down refinement + constraint propagation of VEXED. However, it extends the VEXED approach by concentrating on (i) a multigoal situation of special

significance, in particular how to handle global resource constraints (e.g., gate count) as well as functional goals; and (ii) methods for **automating the control** of design.

One idea embodied in DONTE is to work with **resource budgets** distributed over a candidate design structure, and to focus attention on those parts of the design where the estimated resource use is most critical compared to the budget. Another idea is to do a preliminary, trial, design; to find out how constraints imposed by one decision affect other decisions; and then to redo the design, trying to order decisions in a manner that utilizes best the knowledge about constraint dependencies.

The following is an example of the type of problem that DONTE handles: find an (hardware) implementation for a stack that stores data as a list of 2-bit elements, performs Push and Pop functions, uses TTL circuits, and is constrained to use one control port, and not to exceed a gate count of 60.

The DONTE project continues to provide a focus for research on resource constrained design, and on control of relaxed reduction processes in the environment of digital circuit design tasks.

One aspect of the DONTE approach is conceptually related to the AIR-CYL project at Ohio State [Brown and Chandrasekaran 1985]. In the AIR-CYL project, processes of routine design are studied in the context of designing Air Cylinders. In response to given design specifications, a pre-cached rough design structure is selected, which contains several open parameters. The possible values of the parameters are governed by a given set of constraints. In a subsequent stage of design refinement, the parameters are assigned values through a sequence of decisions that are guided by the constraint structure. This process is similar to the DONTE approach of generating a rough design, and then refining/improving the design by using the information gathered on constraint dependencies between parts of the rough design. However, in the DONTE case, the initial design structure is assembled by the system, and not retrieved as a pre-stored schema.

(iii) REDESIGN [Mitchell et al1983]; BOGART [Mostow 1988]

A very common way of approaching a design task is to focus on a 'similar', previously completed design, and then to use the previous design as a **prototype** which must be appropriately adjusted in order to solve the problem on hand. The design record plays a central role in such a process of redesign.

Early research in this area was carried in the REDESIGN project at Rutgers, which has been succeeded recently by BOGART.

The domain of the REDESIGN project was digital circuit design . Its task was to redesign digital circuits to meet altered functional specifications. The project preceeded VEXED, and it introduced the main representations and knowledge bodies that are being used in VEXED. In particular, the notion of a design record in the form of a design plan, as currently used in VEXED, was a key element in the approach. A design plan is a representation of the design where the design structure is articulated/explained in terms of the sequence of refinements made to generate it. The REDESIGN system had available refirement rules of the type used in VEXED, and the CRITTER constraint propagator. The system took as input the design plan of the original circuit, and it proceeded - via a means-ends analysis - to repair constraint violations that the altered design specifications imposed on the original design. The repairs included insertions of

new interfaces and changes in module specifications. The mode of operation was interactive: the system localizes (sub)modules that need to change, generates redesign options as (sub)module specifications, and ranks options using weak heuristics; the user selects and implements a redesign option; the system detects and repairs side effects, via another redesign.

BOGART is REDESIGN's successor at Rutgers. The system takes a broader view of the redesign task, it tries to reduce the need for user intervention, and it is integrated in VEXED. The goal is to find effective ways of reusing VEXED designs by adapting them to new specifications - which is more general than handling a specification change for a given design. The approach is to use the design plan of a previous relevant design as a guide for the construction of a plan for the current design. More specifically, an attempt is made to use as much as possible of the high level steps in the previous plan (these correspond to the 'large grain' specification of the design structure) for the current design situation. The mode of operation is interactive: the user selects a relevant previous design plan; the system 'replays' successive steps of the plan in a top-down mode, figuring out which new modules correspond to which old ones; when attempts to establish correspondences fail, the full VEXED is used to do the rest of the design.

Future research in this area will be directed to methods for automatically retrieving design plans relevant to a given design problem, finding the corresponding parts of new and old designs, and deciding which parts must be changed. Also, more work is planned on methods for handling the required changes in a design - by specialized patching operations, or by general design approaches.

(iv) LEAP [Mitchell et al 1985b]

The LEAP system is a "learning apprentice" for the VEXED digital circuit design system, whose development started at Rutgers about four years ago. During the operation of VEXED, a user chooses what module to refine next and which implementation rule to apply. A user who doesn't like any of the rules applicable to a module can elect instead to refine it by hand, using a graphic editor. LEAP uses the domain knowledge (theory) on circuit analysis available to VEXED to verify/explain the correctness of the manual step, and then it generalizes the explanation into a new implementation rule. The rule retains only the features of context and function that are mentioned in the explanation.

A prototype of LEAP exists currently, and it has learned several simple rules of digital circuit design. One of its limitations comes from limits on the circuit verifier. Also, since learning in LEAP takes place on basis of a single 'example', via Explanation Based Generalization (EBG) [see Mitchell et al 1986a], it relies on a strong domain theory. This requirement does not pose problems when the learning of rules is at the boolean logic level, but it starts to create difficulties at higher levels of design (i.e, closer to system architecture). Another current limitation of LEAP is its inability to learn under what conditions to apply which rule of implementation, or to provide the user with the information needed how to choose. Further research on LEAP is focusing on these limitations.

In addition to the LEAP approach, where a design system can learn by observing actions of experts, there is another approach to learning that is based on the system's own experience in design. A system could use its experience in order to generalize succesful decisions, avoid unsuccesful decisions and order decisions more effectively. There is a growing activity in the machine learning community on the automation of learning from problem solving experience [see Mitchell et al 1986b]. Work in this area, which is

especially oriented to design problems, is now underway at Rutgers [see Mostow and Bhatnagar 1987]. Progress in methods of learning from problem solving experience or in learning from observation, can have significant impact on the automatic acquisition of design expertise, and more generally, on modes of **guiding the evolution** of realistic design systems. However, much basic AI work is still needed on learning approaches to design.

b. Mechanical Design of Rotational Transmissions: MEET
[Langrana et al 1986]

Research on the MEET project started at Rutgers about three years ago. The MEET system assists in the design of gear, pulley and V-belt systems in response to given functional specifications, domain constraints, and optimality conditions. An important goal of this project is to test and extend the VEXED method of top-down design in a different domain. MEET was implemented in EVEXED, a general system framework that embodies the VEXED design method.

In a manner analogous to functional specifications in VEXED , the functional specifications in MEET are given as relationships between "states of motion" of the input and output of the desired design. A "state of motion" of a mechanical element specifies its rotational speed, its direction of rotation, the power associated with the motion, and the element's location. A design structure (i.e., a solution to the design problem) is represented as an assembly of motion transmitting modules and their linkages.

The design process in MEET has two phases. In the first phase, the VEXED method of top-down design is used. The reasoning proceeds from given functional specifications to a candidate design structure, via a sequence of refinement + constraint propagation steps. The knowledge used for this phase includes (i) a set of implementation rules (e,g., If desire a gear ratio r, where r>10, Then use a compound gear with each having ratio SquareRoot (r)); (ii) a taxonomy of module types (e.g., crossed-belt systems); and (iii) knowledge about functional properties of modules (e.g., If output of a crossed-belt rotates clockwise, then input rotates counterclockwise). At the end of the first phase of design, MEET has a candidate design structure with several open parameters.

The second phase of design is devoted to assigning values to the open parameters of the candidate structure. In this phase, approximate numerical optimization methods are used, such as constrained hill climbing. A typical task in this phase is to assign gear dimensions to a gear module that was roughly specified in the previous design phase. This involves the choice of values for diameter, face and number of teeth of the gear, under given strength constraints and optimality conditions (e.g., min weight, cost).

Although the MEET system incorporates the symbolic method of VEXED for obtaining a qualitative specification of possible design solutions, it needs a different approach for the detailed design of submodules that must satisfy certain physical and optimality constraints. Reasoning with these constraints is best handled by working with conventional mathematical models and numerical methods. Many design tasks require a combination of symbolic approaches and mathematical/numerical methods. MEET provides a good environment for further study of coordination between symbolic reasoning and numerical optimization. This is an area that needs more work.
In general, development of the MEET system, and experimentation in the domain of mechanical transmissions, is still underway.

V. EXAMPLES Oᶠ PLANNED WORK ON NEW DESIGN TASKS

The following is an outline of new design task environments that are being explored at present. Future work in these areas is planned as a collaborative effort involving people at Rutgers and researchers from other institutions.

a. Design of Computer Architectures

Previous work at Berkeley resulted in the development of a CAD tool, called the Advanced Silicon-compiler in Prolog (ASP), which accepts a high level specification of an instruction set architecture (ISA) as input, and produces a VLSI chip design as an output [Despain et al 1987].

Current plans are to pursue a collaborative effort, involving Berkeley and Rutgers investigators, that will focus on the higher level problem of **how to design a computer architecture in response to requirements about the programs that we want to execute on the computer.** The design goals in this case will be in the form of a set of benchmark application programs that must run as fast as possible on the computer architecture, given such implementation constraints as chip area and power.

A solution to the design problem will be in the form of a ISA and its VLSI implementation. Given a candidate ISA, it can be functionally evaluated by executing symbolically the benchmark programs on a process model of the ISA. Also, features of the candidate ISA's implementation, such as chip area, can be evaluated by obtaining, via ASP, the VLSI structure that corresponds to the ISA.

The proposed approach is to derive from the benchmark programs an initial ISA; and then to transform the initial ISA, via a set of operator applications, until a "near optimal" ISA is obtained that satisfies the functional requirements defined by the benchmark programs, and also the constraints on the implementation.

The reasoning needed to generate the initial ISA is expected to be fairly straightforward, assuming that we have available the formal semantics of the programming language in which the benchmark programs are expressed. The main problem is how to transform the initial ISA into a "near optimal" ISA. Here we have a formation problem, where several cycles of 'generate , evaluate, and revise' are needed, and most of the domain knowledge is used in an *a posteriori* mode, in the evaluation phase of each cycle. These problems are more difficult to handle than derivation problems where domain knowledge is used primarily in an *a priori* mode for the generation of candidate designs (such as in VEXED).

However, many realistic design problems are of formation type. Therefore, it is important to explore possible approaches to these problems, by building on top of the work that is slowly accumulating in AI in this area [see Amarel 1986], and by focusing on domains and tasks on which human designers have developed substantial experience so far. Fortunately, the approach proposed here is similar to the approach that has been used to manually create a number of succesful computer architectures. For example, the well known Reduced Instruction Set Computer (RISC) architectures were developed in just such a cycle of executing programs on a proposed architecture, analyzing the results, modifying the architecture , and executing again.

Work in this area can have strong impact on computer design, and in particular on the rapid prototyping of accelerators and other special-purpose processors that are tuned to the efficient solution of special classes of problems. From the AI point of view, the issues elicited by work in this design task include (i) representational choices, (ii) handling of multiple goals, (iii) methods for solving formation problems, and (iv) approaches to the partial inversion of domain knowledge so that it can provide direct guidance to the process of going from one candidate design to the next. Another important issue is **complexity.** By focusing on a design problem of increased complexity (relative to current efforts) we are forced to examine problems that may emerge from attempts to scale up current methods, especially in areas of problem decomposition and constraint propagation.

b. Marine Design

In previous work at SAIC's Marine Hydrodynamis Division, powerful computer-based systems were developed for assisting designers in various tasks of ship and marine platform design. In particular, these systems were used for the **succesful design of the 12-meter yacht Stars & Stripes which won the 1987 America's Cup competition** [Letcher et al 1987]. The goal of this design effort was to generate a hull/keel/winglet configuration for a yacht that would maximize the chances of winning the competition - given a set of assumptions about the sea and weather environment of the races, about the opponent, and about the tactics to be used by the yacht's crew.

The design effort for the 1987 America's Cup competition was unprecedented in both scope and depth in the domain of yacht design. The project produced new aerodynamic and hydrodynamic theories; developed comprehensive computer-based evaluation frameworks for candidate designs, and applied them to the analysis of thousands of hull/keel/winglet configurations; and perfected a methodology for coordinating computer design/evaluation and field construction/testing.

In the course of this effort it became evident that the use of complete hydrodynamic theories to evaluate detailed candidate designs was cumbersome and very costly in time and computer resources. This led to the development of **simplified computational models** based on approximate theories of the physics of flow and on simplified yacht geometries. In addition to facilitating the evaluation of candidate designs, these simplified models enabled the performance of **parametric studies on the functional effect of changes in key structural features of candidate designs.** These computer-based, symbolic, experiments helped to improve understanding of component functions, which in turn helped the process of searching for an optimal design configuration. A similar experience with the advantages of developing and using appropriate specialized and simplified models was obtained during the design of SWATH (Small Waterplane Area Twin Hull) ships at SAIC. In the SWATH design effort, specialized models were used to solve the "inverse problem" of specifying the hull shape from the requirement of optimal total drag.

The main emphasis of these efforts has been on computer-based **analysis and evaluation** of candidate designs - from the point of view of their behavior in the physical environment, and also from the point of view of higher level goals, such as winning a race. The choice of a candidate design and the transition from one candidate to the next - in light of the evidence provided by the evaluation of previous candidates and by parametric experiments - was done by people. The automation of these choices is not

easy, since we are faced with a **formation problem**, where the available knowledge is in a form that allows little direct guidance from design goals to candidate configurations.

However, the experience with manual designs has provided valuable inputs for the development of computer-based approaches to the solution of some of the formation problem encountered in this domain. More specifically, the experience at SAIC has shown (i) the importance of finding appropriate specialized/simplified models, (ii) the significance of carrying out disciplined experiments in the space of designs to obtain an understanding of local structure-function relationships, and (iii) the desirability of using the computer to assist in the overall control of the design process.

Current plans are to pursue a collaborative effort involving researchers from SAIC and Rutgers, which builds on top of the previous design experience, and focuses on the exploration of AI methods that will enable computer-based systems to increase their participation in the high level tasks of generating/modifying candidate solutions. This work will require **close coordination between AI methods and techniques on the one hand, and mathematical models and numerical methods on the other.** There are a number of system issues that must be resolved in order to couple gracefully the numerical packages used in previous 'conventional' design efforts with the system frameworks used in AI. Research will center on computer approaches to the following tasks:

the process of formulating **appropriate simplified models and abstractions** in the domain, and their use in design decisions;

the identification and effective use of **problem decomposition;**

the handling of **multiple goals,** and especially the integrated evaluation of candidate designs relative to the different goals; and

control of the search for solution - with emphasis on **experimentation in design space** to find structure-function regularities, and on methods for using this knowledge to increase the effectiveness of problem solving.

Progress in this domain promises to have a significant impact on innovation in marine design. Also, this project provides an excellent experiential basis for work on key AI issues related to design, and a good testbed for the exploration of new ideas in this area.

c. Aircraft Design

There is a close relationship between problems in aircraft design and problems in marine design. There are however, very important differences. Among these, most notably, is the degree of tolerance or sharpness of the design relative to design constraints. In general, aircraft design is much more constrained.

Recent work at NASA Langley and Lockheed Aircraft [Sobieszczanski-Sobiesky et al 1982, 1984, Sobieszczanski-Sobiesky 1988] has resulted in important methodological advances in certain aspects of aircraft design. In particular, a new systematic approach to multilevel **design decomposition** was proposed. The proposed approach was succesfully demonstrated in the preliminary design of a large transport aircraft. In general, this research elucidated some of the difficult issues involved in selecting a 'good' decomposition of the design process, i.e., one that minimizes coupling between component

processes. Work in the related area of handling **multiple design goals** has resulted in a mathematical framework for concurrent design, which was used to guide such processes as wing design that takes into consideration both aerodynamic and structural specifications.

Current plans are to pursue a collaborative effort, involving researchers from NASA Langley, SAIC Applied Physics Division, and Rutgers, which will build on the previous work in this area, and will focus on the **automation of decomposition processes in aircraft design** via a combination of AI methods and mathematical optimization techniques. Another proposed direction of research is to focus on **concurrent design of aircraft wings,** and to explore ways in which appropriately chosen simplified aerodynamic and structural models can be brought to bear jointly on design decisions. This work will require close synergy between AI methods and numerical computations.

As in the marine design domain, work in the aircraft domain depends heavily on our ability to develop computer-based methods for **finding and using approximate theories and specialized models.** Such simplified models are needed, not only because of the computational complexity involved in using complete theories, but also because they provide a basis for a qualitative understanding of structure-function relationships that can be used to guide design decisions. Current work in AI on **qualitative exploration of dynamic systems** [see Hut and Sussman 1987] is relevant here. More work is needed in this area.

The potential practical impact of progress in the domain of aircraft design is enormous. From the point of view of AI research, the problems are similar to those encountered in marine design, except that the increased tightness of design constraints may induce changes in approach. In general, by studying similar problems in closely related domains we are in a better position to assess the generality and transferability of approaches.

d. Design of Chemical Processes and Materials

The domains of marine and aircraft design that we discussed above are characterized by comprehensive bodies of theory that can be used for predicting properties of candidate designs. However, the theories are essentially **intractable** (very demanding in time and computer resources), and we need to develop approximate, specialized, theories in order to proceed realistically with the process of design.

There are many design domains that do not have a body of theory on which to base reasoning about designs, or they have theories that are **incomplete or innacurate.** This is the case in the design of chemical processes involving poorly characterized reactions. In such cases, the design process can at best produce a plausible design, which must be then **physically implemented and tested,** and after this it must be refined based on analyzing the results of the test. This can be seen as a prototype-test-refine approach to design.

This approach is currently being studied at CMU as part of a project to develop an intelligent assistant for the design, implementation, interpretation, and optimization of reaction processes in a particular branch of organic chemistry. The chemistry focus of this research is the synthesis of new covalently linked multichromophore assemblies, which is expected to help in understanding photosynthesis and other biological processes. The project is concerned with the **synthesis of new organic molecules via new**

69

reaction steps. In particular, knowledge about the reactions is incomplete. For a given reaction step one might know in advance the necessary starting reactants and primary products, but be uncertain about possible side reactions, the effect of various catalysts on the reaction step, or the precise effect of concentration, temperature, pressure, or other parameters on reaction yield.

In such a situation, a first phase of an initial design cycle will have as a goal to generate **a prototypical version** (a sketch) of the desired sequence of reaction steps, with nominal values assigned to reaction parameters. This can be easily done with current AI methods of goal-directed planning, based on whatever initial knowledge exists about individual reaction steps. A second phase will consist of an experiment that implements the initial sequence and determines the actual outcome of the synthesis. Differences between intended and actual outcomes (of individual steps and of the entire sequence) are then established and analyzed. In a third phase, in light of the differences uncovered by the experiment and by using additional basic knowledge about reaction mechanisms, the reaction sequence is **redesigned.** This can be seen as a refinement of the design. At this point, a second design cycle is initiated, similar to the first, and the process continues until a design with the desired properties is actually obtained.

From the point of view of AI research, the problems encountered in this project are mainly in the areas of **knowledge acquisition, refinement, and representation.** A collaborative effort is now planned, involving researchers from CMU and Rutgers, which will focus on the following issues:

> **Characterization of the types and roles of knowledge** that guide (a) initial design and (b) design revisions - in response to discrepancies between design goals (intentions) and actual properties of candidate designs (obtained by observation). In particular, how can design revisions be supported by basic knowledge (underlying principles) in the domain.

> Methods for automatically **refining the initial domain knowledge** in response to new information gained from analysis of experiments in the prototype-test-refine cycles. Automatic modification/improvement of theories about individual reaction steps.

> Methods by which the system may **acquire knowledge** from the user both by direct input and by analyzing the user's behavior in solving individual synthesis problems. Extension of the concept of a 'learning apprentice', which has been developed in the context of digital circuit design [see LEAP in IV.a. above], to the present domain.

Progress in this project will provide useful tools for synthetic chemistry. It will also provide a model for handling a large class of design tasks that are characterized by incomplete or inaccurate domain theories. The **design of new materials** is in this class, and has many similarities with the design of reaction processes for the production of new chemicals. Collaborative work in this area is planned, involving investigators from CMU and Rutgers, with emphasis on the synthesis of metal alloys, fiber optics and composites.

The plan is to develop a data base of prior material designs, including process information and parametric models for extrapolating from prior experience. To the extent that scientific knowledge exists (possibly incomplete) for guiding material synthesis, it will be explicitly used for search of processes that will produce materials with desired characteristics - in a manner similar to the process outlined for chemical

synthesis. If little theory is available in the domain, then design can be guided by the cases recorded in the data base of previous material designs. The design process will proceed **by analogy,** based on one or more recorded cases that are 'similar' to the case under consideration. This is an area where more work is needed. The availability of appropriate representations of records of previous designs is essential for the effectiveness of processes of design by analogy

VI. EXPECTED IMPACT ON THEORY AND PRACTICE OF DESIGN

From the point of view of **Design,** a major push in AI (and in related areas of advanced computing) along the lines discussed above, can be expected to produce the following results:

advanced design and analysis tools to assess the performance, cost, reliability, maintainability, producibility and other attributes of design and manufacturing alternatives;

high level integrated design systems for specific domains (e.g., digital systems, marine platforms, etc.) containing the knowledge necessary to generate design and implementation (manufacturing) plans from functional specifications and other constraints, so that the designs are optimized for manufacturability and maintainability as well as operational performance;

consulting systems for design and process planning which integrate and scale-up results demonstrated in specific domains, and learning capabilities for improving system performance with experience;

generic models of design processes, and software environments based on these models, that support development of automated (or quasi-automated) design systems in a number of domains;

scientific advances and technology innovations in the general area of design and manufacturing.

VII. CONCLUDING COMMENTS

In recent years we witnessed a growing level of activity in AI and Design. Experience drawn from various studies and exploratory systems in this area has been accumulating rapidly. There has been good progress in handling certain kinds of design tasks in domains where relevant knowledge is available in a fairly well structured and tractable form, and at levels of complexity that are relatively modest. What is more important is that the work done so far has resulted in an increased understanding of the key scientific and technical issues involved. In particular, previous work in this area has helped us to identify specific directions of basic work in AI that need increased attention in order to provide the foundations for computer-based handling of a broader range of realistic design problems.

The directions of AI research that need to be vigorously pursued are in the broad area of **problem solving** - with emphasis on complex planning and constraint satisfaction problems. Within this broad area, there are certain issues of reasoning about problem formulations, of control of problem solving processes, of

choice of representations, and of knowledge handling (acquisition, structuring, management and appropriate use) that require special attention. These issues were discussed in some detail in sections II and III above. It is interesting to note that the directions of AI research that are induced from considerations of design problems have a strong overlap with main lines of basic work in AI - as seen from inside the discipline.

Thus a research push in AI and Design can be expected to have significant impact both on the theory and practice of Design and on the scientific and technological advancement of AI in general.

We strongly suggest that projects in the area of AI and Design should include both **design-oriented system developments** that explore AI approaches in specific domains, and **core research in AI** that has general relevance to design tasks. The 'core' work should grow in the environment of the system development efforts, and it should maintain close conceptual links with these efforts. There is ample experience in AI that shows the power of such an approach, e.g., the AI in Medicine projects of the seventies [see Clancey and Shortliffe 1984, Amarel 1974] and the DENDRAL project at Stanford [see Lindsay et al 1980]. As can be seen from the examples of design tasks discussed in previous sections, there are several design domains that can provide excellent focal points for intensified future work along the lines that we are proposing.

We can expect that the 'pull' of a major effort on AI and Design can have an impact on advanced problem solving in AI that is analogous to the impact of the AI in Medicine 'pull' of a decade ago that resulted in major progress on classification and interpretation problems in AI, and in the phenomenal development of the knowledge-based, expert, systems technology.

Since the early seventies there have been substantial developments in 'conventional' CAD/CAM systems, where the emphasis is on tools for representing, analyzing and evaluating designs. These efforts have resulted in technologies and systems that are widely used by industry. Recently, developers of CAD/CAM systems, and designers with an eye on faster innovation cycles, have been looking for computer-based capabilities that increase overall flexibility, capture and retain previous design experience, and provide support in the early, conceptual, stages of design. Such support is needed in the formulation and management of design specifications and constraints, in the generation of design options, and in the control of the multitude of processes that take place during design. In short, they have been looking for **"intelligent" design assistants.** Thus, we see a confluence of attitudes and developments in the 'conventional' CAD/CAM community and in the AI and Design community.

In light of the present state of AI, and of the progress that has been achieved to date in AI and Design, and also taking into consideration the pressures that are building from industry for methods and tools that can increase productivity, we believe it is timely to move towards the implementation of certain parts of the CAP concept/plan which we outlined in section I.c. above, in particular of those aspects that seek to capitalize on developments and opportunities in AI. **The major goal is to obtain dramatic improvements in industrial productivity via effective automation of design and manufacturing processes.** Our general thesis is that the computer field is now at a point where it can provide the intellectual foundations and the technical basis for an effort that can respond effectively to this challenge. The more specific message that this paper intends to convey is that **substantial progress can be made towards the goal of productivity improvement** by strengthening/accelerating research and development in certain   key

areas of AI (and in related areas of advanced computing) that are relevant to the understanding and mechanization of design in its various forms.

REFERENCES

Amarel, S., (1987), 'Problem Solving', in _Encyclopedia of Artificial Intelligence_, Shapiro, S. (ed), John Wiley & Sons, New York, 1987

Amarel, S, (1986), 'Program Synthesis as a Theory Formation Task - Problem Representations and Solution Methods', in Michalski, R., Carbonell, J., Mitchell, T., (eds), _Machine Learning: An Artificial Intelligence Approach_, Vol. II, Morgan Kaufmann, Los Altos, CA 1986.

Amarel, S. (1983), 'Problems of Representation in Heuristic Problem Solving; Related Issues in the Development of Expert Systems', in Groner, R. Groner, M., Bischof, W. (eds), _Methods of Heuristics_, Lawrence Erlbaum, Hillsdale, N.J. 1983.

Amarel, S. (1974), ' Computer-based Modeling and Interpretation in Medicine and Psychology: The Rutgers Research Resource' in _Computers in Life Sciences Research_, Siler, W., Lindberg, D., (ers), Plenum Press, New York, 1974.

Brown, H., Tong, C., Foyster, G. (1983), 'Palladio: An exploratory environment for circuit design' in _IEEE Computer Magazine_, December 1983.

Brown, D. and Chandrasekaran, B. (1985), 'Expert Systems for a Class of Mechanical Design Activity', in Gero, J.S. (ed) _Knowledge Engineering for Computer Aided Design_, North Holland, Amsterdam,1985.

Clancey, W., Shortliffe E. (eds) (1984), _Readings in Medical Artificial Intelligence: The First Decade_, Addison-Wesley Publishing Co., Reading, Mass, 1984

Despain, A., McGeer, P., Bush, W., Cheng, G. (1987), 'An Advanced Silicon Compiler in Prolog' in _Proceedings of International Conference on Computer Design_, Rye Town, N.Y., 1987.

Hut, P., Sussman, G., (1987), 'Advanced Computing for Science', _Scientific American_, October 1987.

Kelly, V. (1985), 'The CRITTER System - An Artificial Intelligence Approach to Digital Circuit Design Critiquing', PhD thesis, Department of Computer Science, Rutgers University, January 1985.

Langrana, N., Mitchell, T., Ramachandran, N. (1986), 'Progress Towards a Knowledge-Based Aid for Mechanical Design', in _Proceedings of the Symposium on Integrated and Intelligent Manufacturing_, ASME Winter Annual Meeting, Anaheim, CA, 1986.

Letcher, J., Marshall, J., Oliver, J., Salvesen, N. (1987), 'Stars & Stripes' , _Scientific American_, August 1987.

Lindsay, R., Buchanan, B., Feigenbaum, E., Lederberg, J. (1980), _Applications of Artificial Intelligence for Organic Chemistry: The Dendral Project_, Mc Graw-Hill, New York, 1980.

Mitchell, T., Keller, R., Kedar-Cabelli, S. (1986a) 'Explanation-Based Generalization: A Unifying View', Machine Learning 1(1), 1986.

Mitchell, T., Carbonell, J., Michalski, R. (eds) (1986b), Machine Learning: A Guide to Current Research, Kluwer Academic Publishers, Boston 1986. Based on the Third International Machine Learning Workshop, Skytop, Pennsylvania.

Mitchell, T., Steinberg, L, Shulman, J. (1985a), 'A Knowledge-Based Approach to Design' in IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-7(5), September 1985.

Mitchell, T., Mahadevan, S., Steinberg, L. (1985b), 'LEAP: A Learning Apprentice for VLSI Design' in Proceedings of IJCAI-85, Los Angeles, CA, August 1985.

Mitchell, T., Steinberg, L., Kedar-Cabelli, S., Kelly, V., Shulman, J., Weinrich, T., (1983) 'An Intelligent Aid for Circuit Redesign' in Proceedings of AAAI-83, AAAI, August 1983.

Mostow, J. (1988), 'Design by Derivational Analogy: Issues in the Automated Replay of Design Plans', Artificial Intelligence, 1988 (to appear in special issue on Machine Learning); also available as Rutgers AI/Design Project Working Paper No. 80, December 1987.

Mostow, J., Bhatnagar, N. (1987), 'Failsafe -- a floor planner that uses EBG to learn from its failures' in Proceedings IJCAI-87, August 1987.

Press, F. (1987), in 'A High Technology Gap', Council on Foreign Relations, pp. 14-15, New York, 1987.

Simon, H.A. (1981), The Sciences of the Artificial, The MIT Press, Cambridge, Mass, 1981 (the first edition appeared in 1969).

Sobieszczanski-Sobiesky, J., (1988), 'On the Sensitivity of Complex, Internally Coupled Systems', AIAA Paper 88-2378, presented at AIAA/ASME/ASCE/AHS 29th Structures, Structural Dynamics and Materials Conference, Williamsburg, VA, April 1988.

Sobieszczanski-Sobiesky, J., Barthelemy, J-F., Giles, G., (1984), 'Aerospace Engineering Design by Systematic Decomposition and Multilevel Optimization', NASA Technical Memo. 85823, NASA LRC, June 1984.

Sobieszczanski-Sobiesky, J., Barthelemy, J-F., Riley, K. (1982) 'Sensitivity of Problem Solutions to Problem Parameters', AIAA Paper 81-0548R, AIAA Journal, Vol. 20, No. 9, September 1982.

Steinberg, L. (1987), 'Design as Refinement Plus Constraint Propagation: the VEXED Experience' Proceedings AAAI-87, July 1987.

Tong, C. (1988) 'Knowledge-Based Circuit Design', PhD Thesis, Computer Science Department, Stanford University, May 1988.

Tong, C. (ed) (1987a) , Special Issue on AI in Engineering Design, International Journal of Artificial Intelligence in Engineering 2 (3), July, 1987.

Tong, C. (1987b), 'Towards an Engineering Science of Knowledge-Based Design, International Journal of Artificial Intelligence in Engineering 2(3), special issue on AI in Engineering Design, July 1987.

Tong, C., Sriram, D. (1988), Artificial Intelligence Approaches to Engineering Design, Addison-Wesley, 1988 (to appear).

Toward a New Era in US Manufacturing: The Need for a National Vision, (1986), by the Manufacturing Studies Board, Commission on Engineering and Technical Systems, National Research Council, National Academy Press, Washington D.C., 1986.

Young, J. A. (1988), 'Technology and Competitiveness: A Key to the Economic Future of the United States', Science, Vol 241, pp. 313-316, 15 July 1988.

July 31, 1988

Recovery of 3-D Motion and
Structure from Image Correspondences
Using a Directional Confidence Measure

Gilad Adiv
Edward Riseman

COINS TR 88-105

December 1988

RECOVERY OF 3-D MOTION AND STRUCTURE

FROM IMAGE CORRESPONDENCES

USING A DIRECTIONAL CONFIDENCE MEASURE

Gilad Adiv*
Edward Riseman

ABSTRACT

We present a new scheme for computing 3-D motion and structure from a flow field representing either image velocities or image displacements between two frames. This scheme is based on a global least-squares technique, introduced in [Adi85a,b], for minimizing the deviation between the given flow field and the field predicted by the hypothesized 3-D motion and structure. Here, this technique is generalized by assigning a *directional confidence measure* to each flow vector. This confidence measure is defined by two orthogonal axes and corresponding confidence values, representing the reliability of the estimated image motion along each axis. It is shown how to relate these confidence values to the error distributions of the estimated flow values. The directional confidence measure is especially useful for recovering 3-D information from correspondences of line segments or edge points, where the normal component of the image motion is much more reliable than the tangential component. Experiments based on simulated and real data demonstrate the improvement achieved by employing a directional confidence measure instead of a scalar confidence measure. Finally, we show that the reliability of depth estimates can be predicted from the confidence measure.

---

* The author is with Rafael, POB 2250(34), Haifa 31021, Israel. Most of this work was performed when he was a visiting scientist at the Computer and Information Science Department, University of Massachusetts, Amherst, MA 01003.

1. INTRODUCTION

The problem of passive navigation, where a sensor is moving through a stationary environment, is one of the major research issues in the area of dynamic visual interpretation. Given two perspective views from such a sensor, it is possible to extract the 3-D motion of the sensor and the structure of the environment, up to a scaling factor. Such information can be used to control the motion of vehicles or robots.

The most common approach for the analysis of visual motion is based on two phases. The first phase is computation of image correspondences, usually referred to as an optical flow field, or a displacement field. The second phase consists of an interpretation of this field. Many of the algorithms described in the literature use point correspondences in the second phase (e.g., [Ull79], [Lon81], [Bru81], [Tsa84], [Adi85a,b]). Given an image point, we know that it is the projection of one of an infinite number of points in the 3-D space, all of them located on a ray defined by the image point and the lens center. The correspondence of a point in the first image to a point in the second image means that the two 3-D rays associated with these points intersect each other. This puts a constraint on the problem and, therefore, given a sufficient number of point correspondences, the 3-D motion and structure can be extracted (up to a scaling factor).

Recently, a few authors have proposed to compute 3-D motion and structure from line correspondences (e.g., [Liu88], [Fau87], [Spe87]), utilizing the information given by the orientation and the distance from the origin of the lines. This new approach may be very useful in man-made environments where straight lines are dominant and stable features. It has been found, however, that correspondence of a line in two frames does not sufficiently constrain the problem; that is, the 3-D motion and structure can not be

recovered from such information. To understand this, notice that a line in the image is associated with a plane in the 3-D space containing all the 3-D lines possibly generating the image line. A correspondence of a line in the first image to a line in the second image is equivalent, therefore, to the intersection of the two associated 3-D planes. Unfortunately, two arbitrary planes generally intersect each other and, therefore, no constraint on the motion parameters can be obtained from such an intersection. Thus, line correspondences over three frames are necessary for recovery of 3-D motion.

In all interesting applications measurements of image motion are corrupted by noise. Therefore, the recovery of 3-D motion and structure should be based on the minimization of some error function of these 3-D variables. Such a function is usually the sum of error terms, where each term is associated with one image correspondence. The contribution of this term to the global error function should depend on the reliability of the related image motion measurement. In [Bru81] and [Adi85a,b] the overall reliability of each flow vector is assumed to be estimated and represented by a *scalar confidence measure*. This measure was integrated into a least-squares scheme for minimizing the sum of deviations between the measured flow vectors and the corresponding vectors predicted by the hypothesized 3-D parameters.

Anandan [Ana87, Ana88] has introduced a more general confidence measure, which we call the *directional confidence measure*. This measure can be employed as a tool for improving the representation of knowledge about uncertainties of image motion measurements. It is defined by two orthogonal axes and corresponding confidence values, giving the reliability of the estimation of displacement along each axis. Typically, the axis with maximal confidence value will be oriented in the direction of the image gradient. Anandan has applied such a directional confidence measure to the estimation of a dense displace-

ment field. In this technique, each displacement vector is assigned a directional confidence measure, based on the curvatures of an error surface associated with the measurements for determining this vector. The confidence measure is employed to control the smoothing between adjacent vectors. A similar "oriented smoothness" approach is taken by Nagel and Enkelmann [Nag86], but without recognizing the implicit use of a directional confidence measure.

We will employ the directional confidence measure as a tool for developing a unified approach for solving 3-D motion and structure from point and line correspondences. This tool is especially important in the case of line correspondences, and we will use this case for motivating our approach. We have already concluded that line correspondences over three frames are apparently necessary for recovering 3-D motion. Using a third frame is roughly equivalent to using second-order time derivatives of the line parameters. However, such derivatives can not be expected to be recovered reliably in the presence of noise, and this solution may be particularly sensitive to noise if the t' ree viewpoints are close to each other.

In this paper we present another approach. Usually, endpoints of lines in the image can be extracted, and the lines are given as line *segments*. We argue that, utilizing the information given by the location of line endpoints, the 3-D motion and structure can be estimated reliably using only *two* frames. In other words, we will introduce a method for recovering 3-D interpretation consistent not only with the line equations, but also with the location of the endpoints along the line (Fig. 1). This approach can also be regarded as a specific case of solving motion and structure from point correspondences.

Of course, the determination of an endpoint location along a line may be a difficult
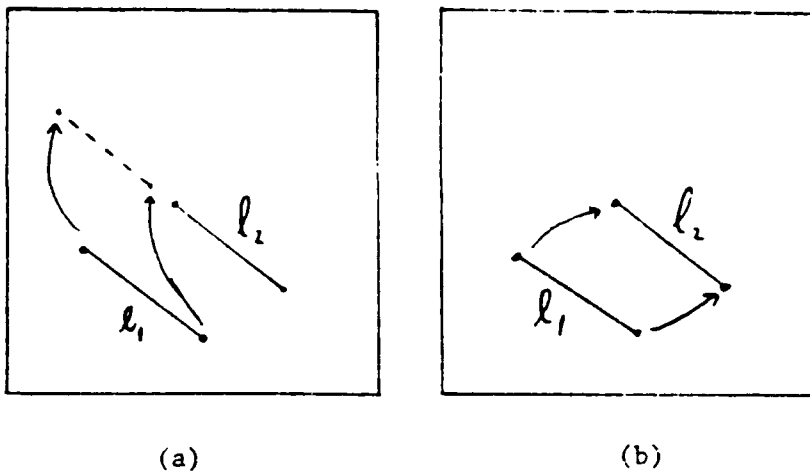
Fig. 1: Correspondence of lines with and without endpoint correspondence. l_1 is a line segment in the first frame and l_2 is a corresponding line segment in the second frame. (a) A 3-D solution that transforms l_1 as shown is supported by this 2-D line correspondence if consistency of the line equations is the only criterion. It is not supported if, in addition, an overlapping of the line segments is required. (b) In this more restrictive sense, a 3-D solution that transforms the endpoints of l_1 to the endpoints of l_2 is maximally supported by the line correspondence.

task, and sensitive to noise. On the other hand, the transverse location of the endpoint can be expected to be measured accurately. Therefore, when evaluating the consistency of a hypothesized 3-D solution with image correspondences of line segments, the deviation along a line should be allowed to be larger than the deviation in the transverse direction (see Fig. 2).

This observation can be given a mathematical formulation by giving the longitudinal deviation a relatively small weight, while giving the transverse deviation a relatively large weight. In other words, the directional confidence measure is suitable for representing our knowledge about the uncertainty of an endpoint location. This approach was already demonstrated by Wells [Wel87] in a constrained case, where the motion is known and the goal is to recover the location of 3-D line segments projected on a sequence of images.

33

(a)   (b)
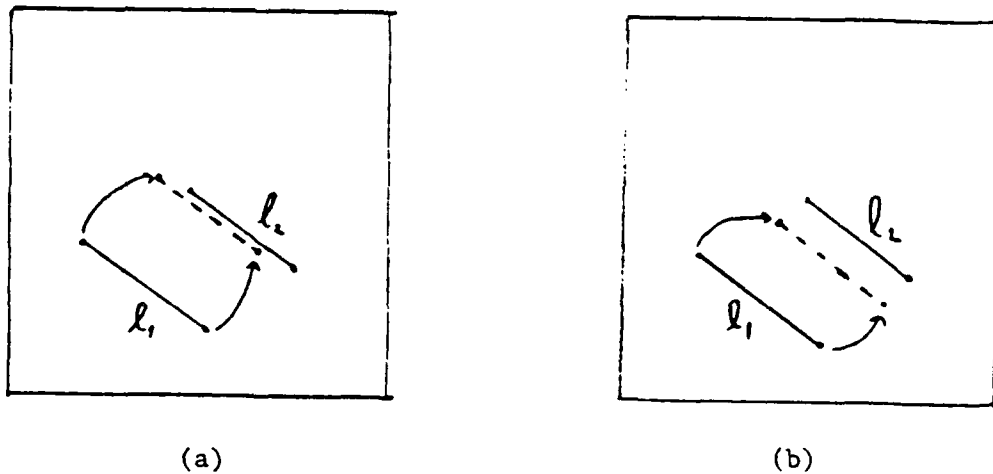
Fig. 2: Uncertainty in line segment position. The uncertainty of the line segment position in the longitudinal direction is much larger than the uncertainty in the transverse direction. (a) Correspondence of line segments l_1 and l_2 is consistent with this uncertainty and, therefore, it supports the realted 3-D transformation. (b) Correspondence is inconsistent with uncertainty in line segment position. Thus, it does not support the realted 3-D transformation.

In the following sections we will develop a general scheme for using a directional confidence measure. As has already been noted, such a scheme is especially needed in the case of line segment correspondences, but it may also improve the results in other cases when 3-D information must be extracted from feature correspondences or optical flow. Given, for example, corner correspondences, one may want to give a higher confidence to the direction perpendicular to the bisector of the angle of an acute corner. Finally, notice that this scheme is relevant not only to motion analysis, but also to stereoscopic vision and image matching.

2. A MATHEMATICAL FORMULATION

2.1 Relating Image Motion to 3-D Motion and Structure

In this section we show how the motion of image features is related to the 3-D camera motion and the 3-D environmental structure, assuming a perspective projection. The camera motion is allowed to be general, with six degrees of freedom, but the environment is assumed to be stationary in this treatment.

Let (X, Y, Z) represent a cartesian coordinate system which is fixed with respect to the camera (see Fig. 3), and let (x, y) represent a corresponding coordinate system of a planar image. The focal length, from the nodal point O to the image, is assumed to be known. It can be normalized to 1 without loss of generality. Thus, the perspective projection (x, y) on the image of a point (X, Y, Z) in the environment is:

$$x = X/Z, \qquad y = Y/Z. \tag{1}$$

The motion of the camera between two time instances, t and t', can be decomposed into two components: rotation $\underline{\Omega} = (\Omega_X, \Omega_Y, \Omega_Z)$ about an axis through the origin, followed by translation $\underline{T} = (T_X, T_Y, T_Z)$. If (X, Y, Z) and (X', Y', Z') are the coordinates at times t and t', respectively, of a point in the environment, then

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = R \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} - \underline{T}, \tag{2a}$$

where the rotation matrix R can be approximated, assuming small values of the rotation
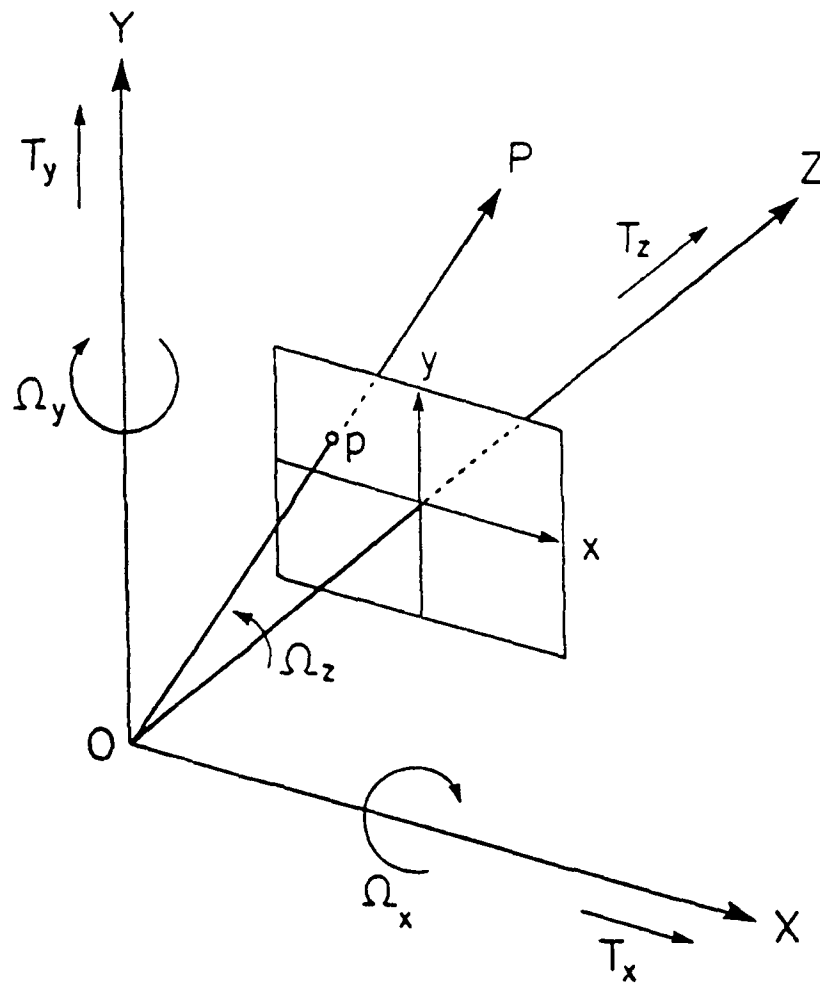
Fig. 3: Coordinate systems. A coordinate system (X, Y, Z) attached to the camera, and the corresponding image coordinates (x, y). The image position $\underline{p}$ is the perspective projection of the point $\underline{P}$ in the environment. $\underline{T} = (T_X, T_Y, T_Z)$ and $\underline{\Omega} = (\Omega_X, \Omega_Y, \Omega_Z)$ represent the translation and rotation of the camera.

parameters, by

$$R = \begin{pmatrix} 1 & \Omega_Z & -\Omega_Y \\ -\Omega_Z & 1 & \Omega_X \\ \Omega_Y & -\Omega_X & 1 \end{pmatrix}. \tag{2b}$$

Now, let (x, y) and (x', y') be the image points corresponding to (X, Y, Z) and

36

(X', Y', Z'), respectively, and let (α, β) be the displacement vector $(x' - x, \ y' - y)$. Then, from Eqs. (1) and (2) we get:

$$\alpha = X'/Z' - X/Z = \frac{1}{Z}\left(x'Z - X\right) =$$

$$= \frac{1}{Z}\left[x'(Z' - \Omega_Y X + \Omega_X Y + T_Z) - (X' - \Omega_Z Y + \Omega_Y Z + T_X)\right] =$$

$$= \Omega_X x'y - \Omega_Y(1 + xx') + \Omega_Z y + (-T_X + T_Z x')/Z. \qquad (3a)$$

Similarly, we can obtain:

$$\beta = \Omega_X(1 + yy') - \Omega_Y xy' - \Omega_Z x + (-T_Y + T_Z y')/Z. \qquad (3b)$$

These equations were previously introduced by Medioni and Yasumoto [Med85]. Notice that

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha_R \\ \beta_R \end{pmatrix} + \begin{pmatrix} \alpha_T \\ \beta_T \end{pmatrix}, \qquad (4a)$$

where (α_R, β_R) and (α_T, β_T) are, respectively, the rotational and translational components of the displacement field:

$$\begin{pmatrix} \alpha_R \\ \beta_R \end{pmatrix} = \begin{pmatrix} x'y \\ 1 + yy' \end{pmatrix}\Omega_X + \begin{pmatrix} -1 - xx' \\ -xy' \end{pmatrix}\Omega_Y + \begin{pmatrix} y \\ -x \end{pmatrix}\Omega_Z, \qquad (4b)$$

$$\begin{pmatrix} \alpha_T \\ \beta_T \end{pmatrix} = \begin{pmatrix} -T_X/Z \\ -T_Y/Z \end{pmatrix} + \begin{pmatrix} x' \\ y' \end{pmatrix}T_Z/Z. \qquad (4c)$$

As can easily be verified, if x' and y' are replaced by x and y, respectively, then Eqs. (4) express the relations between image *velocities* (α, β) and spatial *velocities*

$(\Omega_X, \Omega_Y, \Omega_Z)$ and (T_X, T_Y, T_Z). In this case, the assumption of small rotation parameters is no longer needed. In the rest of this paper, the term 'flow' refers to both 'displacement' and 'velocity'.

Our basic goal is to extract the motion parameters $\underline{T}$, $\underline{\Omega}$ and the depth values $\{Z\}$ from the flow vectors $\{(\alpha, \beta)\}$, using the relations (4). It is easy to see, however, that $\underline{T}$ and $\{Z\}$ can only be determined up to a scaling factor. Therefore, we will introduce new parameters which represent the extractable information.

Let r be the magnitude of the translation. Assuming that r is non-zero, we define new parameters which are possible to estimate:

$$\underline{U} = \underline{T}/r \tag{5}$$

and

$$\tilde{Z} = r/Z. \tag{6}$$

$\underline{U} = (U_X, U_Y, U_Z)$ is a unit vector, representing the direction of the 3-D translation, and $\tilde{Z}$ represents a normalized version of the reciprocal depth, which we find more convenient to estimate and analyze than Z/r. Employing these normalized parameters, Eq. (4a) can be rewritten as

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha_R \\ \beta_R \end{pmatrix} + \begin{pmatrix} \alpha_U \\ \beta_U \end{pmatrix} \tilde{Z}, \tag{7}$$

where

$$\begin{pmatrix} \alpha_U \\ \beta_U \end{pmatrix} = \begin{pmatrix} \alpha_T \\ \beta_T \end{pmatrix} / \tilde{Z} = - \begin{pmatrix} U_X \\ U_Y \end{pmatrix} + \begin{pmatrix} x' \\ y' \end{pmatrix} U_Z. \tag{8}$$

Let us assume that each flow vector is assigned a confidence measure. In the past we used a measure represented by a scalar, W, giving the overall reliability of the flow estimate [Ana84], [Adi85a,b]. A more general approach is to use a directional quantity, represented by two orthogonal axes and corresponding confidence measures. Along one axis the confidence, denoted by W_t, is maximal, while along the other axis the confidence, denoted by W_l, is minimal. The angle between the axis of maximal confidence and the x-axis is given by ρ ($0 \le \rho < 180°$). Geometrically, the scalar measure can be represented by a circle with radius W, while the directional measure can be represented by an ellipse with a long axis W_t and a short axis W_l (see Fig. 4).

flow vector

W

(a)

flow vector

W_t

ρ

$\to X$ (x'')

W_ℓ

(b)
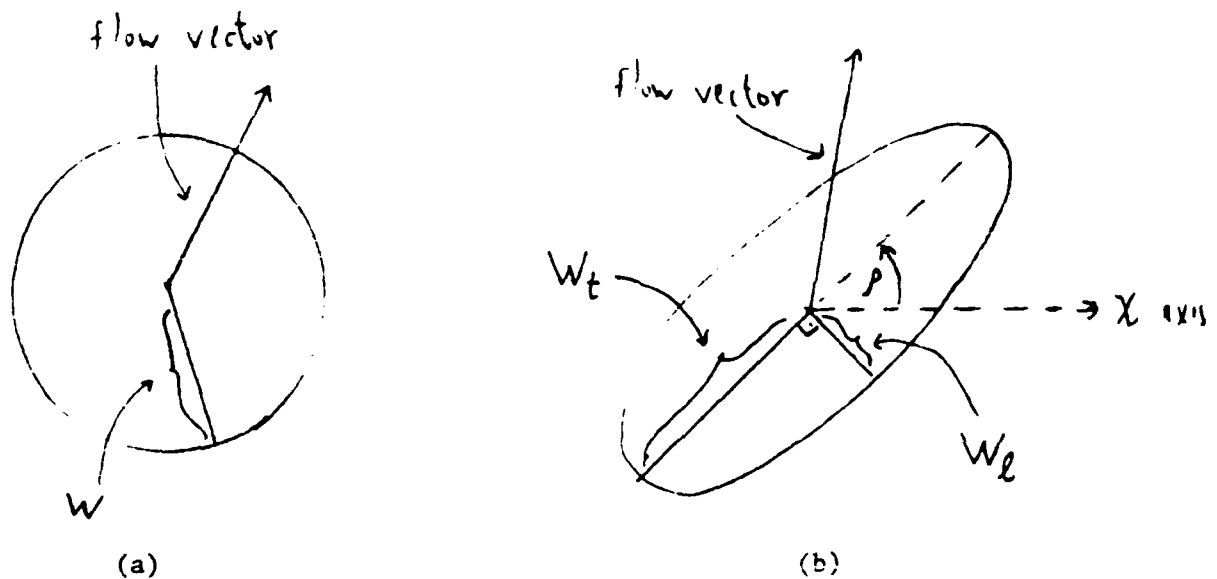
Fig. 4: Geometrical interpretation of scalar and directional confidence measures. (a) The circle represents a scalar confidence measure, where the confidence is uniform with respect to the direction. ((b) The ellipse represents a directional confidence measure, where the confidence varies as a function of the direction.

A directonal confidence measure is computed in ˌAna87, Ana88ˌ for a dense displace-ment field. Typically, in uniform regions both minimal and maximal confidence values are low, whereas at edges (except occlusion boundaries) the confidence is high along the gradient direction and low along the edge, and finally at corners both values are high.

The confidence measure (either scalar or directional) can be used for weighting the contribution of the flow vector to the determination of 3-D motion and structure parame-ters. In order to save computation, it is also possible to select and use a given number of "best" flow vectors, while ignoring the other vectors.

3. A GLOBAL OPTIMIZATION APPROACH USING
A SCALAR CONFIDENCE MEASURE

Before turning to the directional confidence measure, we show in this section how knowledge, represented by a *scalar* confidence measure, can be integrated into a least squares scheme for extracting 3-D motion and structure from optical flow. Let $(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n)$ be n flow vectors measured at the image points $(x_1, y_1), \ldots, (x_n, y_n)$ and assigned scalar confidence values $W_1, \ldots, W_n$. The goal is to extract 3-D motion pa-rameters, $\underline{U}$ and $\underline{\Omega}$, and normalized depth values, $\tilde{Z}_1, \ldots, \tilde{Z}_n$, which are maximally consistent with the available data.

Let us briefly review the approach in ˌBru81ˌ and ˌAdi85a,bˌ, where a least squares scheme is employed. This approach, which is attractive because of its relative robustness to noise, is based on minimizing the deviation between the measured flow vectors and those predicted from the estimated 3-D motion parameters and depth values. The deviation related to each flow vector is weighted by the corresponding confidence value. In other

90

words, we want to find $\underline{U}$, $\underline{\Omega}$ and $\bar{Z}_1,\ldots,\bar{Z}_n$ such that the error function

$$\sum_{i=1}^{n} W_i \left[(\alpha_i - \alpha_{R_i} - \alpha_{U_i}\bar{Z}_i)^2 + (\beta_i - \beta_{R_i} - \beta_{U_i}\bar{Z}_i)^2\right] \tag{9}$$

is minimized (see Eq. (7)). In addition, the constraints $\bar{Z}_i \geq 0$, $i = 1,\ldots,n$, should be satisfied, but, for the sake of brevity, we ignore them in the current discussion. The interested reader is referred to [Adi85a,b].

Given the values of the motion parameters, the optimal value of $\bar{Z}_i$, $1 \leq i \leq n$, can be found by minimizing the corresponding term in the error function (9):

$$\bar{Z}_i = \frac{(\alpha_i - \alpha_{R_i})\alpha_{U_i} + (\beta_i - \beta_{R_i})\beta_{U_i}}{\alpha_{U_i}^2 + \beta_{U_i}^2}. \tag{10}$$

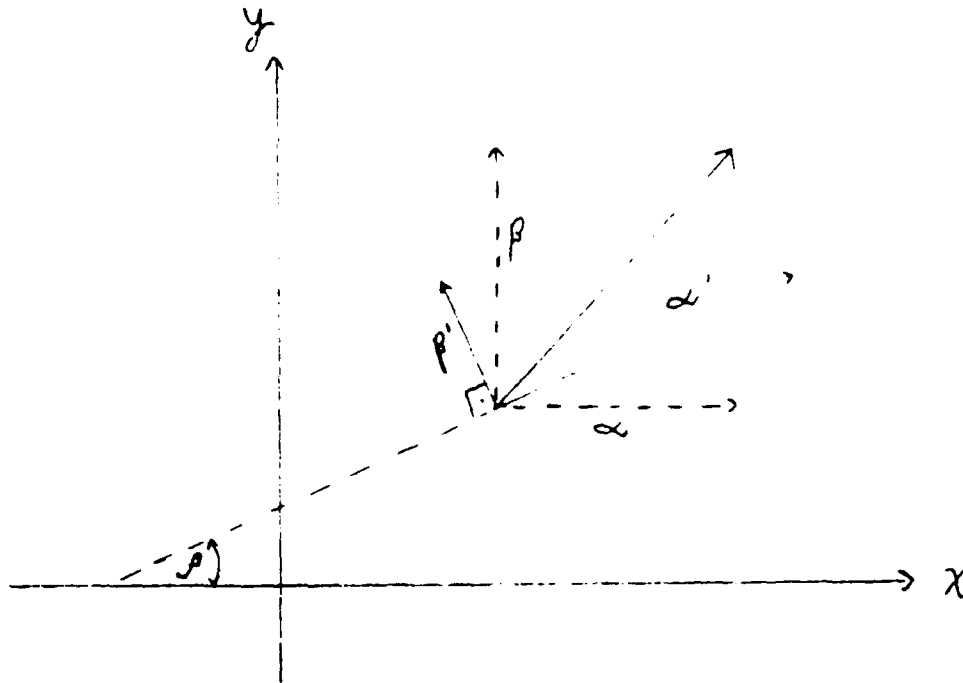Substituting (10), for any $1 \leq i \leq n$, into (9) and expanding the resulting expression yields the following representation of the error, as a function of the motion parameters:

$$E(\underline{U},\underline{\Omega}) = \sum_{i=1}^{n} W_i \frac{[(\alpha_i - \alpha_{R_i})\beta_{U_i} - (\beta_i - \beta_{R_i})\alpha_{U_i}]^2}{\alpha_{U_i}^2 + \beta_{U_i}^2}. \tag{11}$$

The motion parameters are recovered in [Adi85a,b] by deriving from (11) an error measure which correpsonds to possible values of $\underline{U}$. For each hypothesized $\underline{U}$, the optimal rotation parameters and a related error value are computed by solving three linear equations. A minimum value of the resulting error function is determined, using a multi-resolution sampling technique.

4. A GLOBAL OPTIMIZATION APPROACH USING

A DIRECTIONAL CONFIDENCE MEASURE

4.1 Error Functions and a Search Procedure

In this section we generalize the analysis of the previous section by assuming that a *directional* confidence measure is assigned to each flow vector. Let $(W_{t_i}, W_{l_i}, \rho_i)$ represent the directional confidence corresponding to the measured flow vector (α_i, β_i), $1 \leq i \leq n$ (see Section 2.2). In order to weight correctly the deviation between the measured and predicted flow vectors, a rotated coordinate system is separately determined for each vector. using ρ_i as the angle of rotation. Values in a rotated coordinate system will be denoted by the symbol "$'$", e.g. (see Fig. 5):

$$\begin{pmatrix} \alpha'_i \\ \beta'_i \end{pmatrix} = \begin{pmatrix} \cos \rho_i & \sin \rho_i \\ -\sin \rho_i & \cos \rho_i \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix}. \tag{12}$$

Following Eq. (9), the error function to be minimized is

$$\sum_{i=1}^{n} [W_{t_i}(\alpha'_i - \alpha'_{R_i} - \alpha'_{U_i} \bar{Z}_i)^2 + W_{l_i}(\beta'_i - \beta'_{R_i} - \beta'_{U_i} \bar{Z}_i)^2] \tag{13}$$

Again, we can find the optimal value of $\bar{Z}_i$, as a function of the motion parameters, by minimizing the corresponding term in the error function. This can be done by examining the first derivative of (13), with respect to $\bar{Z}_i$, and setting it equal to 0. Thus, we get

$$\bar{Z}_i = \frac{W_{t_i}(\alpha'_i - \alpha'_{R_i})\alpha'_{U_i} + W_{l_i}(\beta'_i - \beta'_{R_i})\beta'_{U_i}}{W_{t_i}\alpha'^2_{U_i} + W_{l_i}\beta'^2_{U_i}}. \tag{14}$$

Substituting (14), for any $1 \leq i \leq n$, into (13) and expanding the resulting expression

Fig. 5: Rotating a coordinate system via the confidence vector. The flow vector (α,β) is given in the image coordinate system (x,y). The angle ρ corresponds to the axis with maximal confidence. It defines a rotated coordinate system in which α' and β' are the new flow values.

yields the following representation of the error, as a function of the motion parameters:

$$E(\underline{U},\underline{\Omega}) = \sum_{i=1}^{n} \frac{W_{t_i} W_{l_i} [(\alpha_i' - \alpha'_{R_i})\beta'_{U_i} - (\beta_i' - \beta'_{R_i})\alpha'_{U_i}]^2}{W_{t_i}\alpha_{U_i}'^2 + W_{l_i}\beta_{U_i}'^2}. \tag{15}$$

The search for optimal $\underline{U}$ and $\underline{\Omega}$ can be based on the search procedure outlined in the previous section. The values of α_i' and β_i', $1 \le i \le n$, can be determined by applying Eq. (12). Similarly, the coefficients of the rotation parameters in α'_{R_i} and β'_{R_i}, $1 \le i \le n$, can be determined from the corresponding coefficients in α_{R_i} and β_{R_i} (see Eq. (4b)). For a given $\underline{U}$, the values of α'_{U_i} and β'_{U_i}, $1 \le i \le n$, can be computed from α_{U_i} and β_{U_i} (see Eq. (8)). Thus, for each hypothesized value of $\underline{U}$, the problem becomes a least squares

93

problem with expressions which *linearly* depend on Ω_X, Ω_Y and Ω_Z. These rotation parameters and a corresponding error measure can, therefore, be computed by solving three linear equations. Thus, an error function, defined on the unit sphere, is obtained. As in Section 3, this function can be minimized using a multi-resolution sampling technique.

4.2 Discussion

A few interesting observations can easily be made from Eqs. (14) and (15):

1) Given a flow vector (α, β) for which $W_l \ll W_t$ (e.g., a point along an edge but not at a corner), one can estimate the corresponding depth by

$$\tilde{Z} \approx \frac{\alpha' - \alpha'_R}{\alpha'_U}, \qquad (16)$$

unless ${\alpha'_U}^2 \ll {\beta'_U}^2$. This estimate is only based on the one reliable component of the flow vector. If, for example, we deal with a line correspondence, then the transverse component of the line displacement will be the dominant one in determining the depth, unless this component is much smaller than the longitudinal component of the displacement.

2) If $W_l = 0$, then, according to Eq. (15), the corresponding flow vector gives no constraint on the optimal motion parameters. This is consistent with the observation already made in the literature that line correspondences in two frames do not constrain the problem. However, assuming that the motion parameters are known (e.g., via the constraints from the other flow vectors in the optimization process), the corresponding depth value (see (16)) may still be recovered.

3) Assuming that $0 < W'_l \ll W_l$, the related error measure is

$$e \approx \frac{W_l}{\alpha'^2_U} [(\alpha' - \alpha'_R)\beta'_U - (\beta' - \beta'_R)\alpha'_U]^2. \qquad (17)$$

Thus, the contribution e of a flow vector to the total error measure is principally determined by the value of W_l. However, even if W_l is small, e may be large if $|\alpha'_U| \ll |\beta'_U|$. Given a line correspondence, for example, this means that the hypothesized focus of expansion (FOE) is along the line. In this case, we have

$$e \approx W_l \left(\frac{\beta'_U}{\alpha'_U}\right)^2 (\alpha' - \alpha'_R)^2. \qquad (18)$$

Therefore, in order to minimize e, α'_R should be close to α'. In the case of a line correspondence, the transverse component of the motion predicted by the hypothesized rotation should be similar to the transverse component of the measured displacement.

Suppose now that line correspondences are determined and an FOE is hypothesized such that there exist lines approximately oriented towards it. Applying the previous discussion, we can check whether there exist rotation parameters consistent with the transverse displacements of these lines and, thus, either compute the rotation parameters or refute the hypothesis. For example, it is possible to compute the rotation parameters of a sensor moving along a road by using the boundary lines of the road.

5. RELATING CONFIDENCE MEASURES TO NOISE DISTRIBUTIONS

In this section we show how confidence values can be derived from probabilistic estimates of measurement errors of flow vectors. Suppose that each flow vector is corrupted by an additive noise with two orthogonal components, N_t and N_l. It is assumed that the expectations of N_t and N_l are 0 and that their standard deviations, σ_t and σ_l, satisfy $0 < \sigma_t \leq \sigma_l$. The angle, denoted by ρ, between the axis corresponding to N_t and the x-axis may be different for each flow vector ($0° \leq \rho < 180°$). Following the analysis and notations in Section 4.1, a coordinate system rotated by ρ_i, $1 \leq i \leq n$, is introduced for each flow vector (α_i, β_i), and the corresponding values are denoted by the symbol '''.

Employing the least squares scheme, it is desirable to normalize each deviation by the expected value of the related measurement error. Hence, the error function to be minimized should be

$$\sum_{i=1}^{n} \left[\left(\frac{\alpha_i' - \alpha_{R_i}' - \alpha_{U_i}' \bar{Z}_i}{\sigma_{t_i}} \right)^2 + \left(\frac{\beta_i' - \beta_{R_i}' - \beta_{U_i}' \bar{Z}_i}{\sigma_{l_i}} \right)^2 \right]. \tag{19}$$

Thus, each deviation is measured in units of the standard deviation of the related measurement error, and the penalty for the deviation is determined by this normalized value. Notice that Eq. (19) leads us to Eq. (13) with

$$W_{ti} = 1/\sigma_{ti}^2, \qquad W_{li} = 1/\sigma_{li}^2. \tag{20}$$

In the framework of the least squares technique with a scalar confidence measure, the deviations are computed in the x and y axes. Let N_x and N_y be the corresponding

96

measurement errors. Their standard deviations, σ_z and σ_y, satisfy the equalities

$$\sigma_z^2 - \sigma_y^2 = \mu(N_z^2 - N_y^2) = \mu(N_t^2 - N_l^2) = \sigma_t^2 - \sigma_l^2. \qquad (21)$$

where μ denotes expectation. in a probabilistic sense. In addition. N_z and N_y are equally distributed and, therefore,

$$\sigma_z^2 = \sigma_y^2 = (\sigma_t^2 - \sigma_l^2)/2 . \qquad (22)$$

For estimating the 3-D motion and structure, we should minimize the expression

$$\sum_{i=1}^{n} \left[\left(\frac{\alpha_i - \alpha_{R_i} - \alpha_{U_i} \bar{Z}_i}{\sigma_z} \right)^2 + \left(\frac{\beta_i - \beta_{R_i} - \beta_{U_i} \bar{Z}_i}{\sigma_y} \right)^2 \right]. \qquad (?3)$$

Using Eq. (22), this leads us to Eq. (9) with

$$W_i = 2/(\sigma_{ti}^2 + \sigma_{li}^2). \qquad (24)$$

The definitions (20) and (24) of W_t, W_l and W yield th$\sim$ following relation between the directional and scalar confidence measures:

$$W = \frac{2}{\sigma_t^2 + \sigma_l^2} = \frac{2}{1/W_t + 1/W_l} = \frac{2W_t W_l}{W_t + W_l} . \qquad (25)$$

This relation will be employed in Experiment 2 for obtaining a scalar confidence measure out of the given directional confidence measure.

6. A CONFIDENCE MEASURE FOR DEPTH ESTIMATES

Many experimental results show that depth estimates are often inaccurate (see, for example, the careful study in [Dut88]). This problem is inherent near the FOE or when

the translation is small relative to the distance of the camera from the observed surface [Adi89]. It is important, therefore, to define a confidence measure for the depth estimates, thus making it possible to distinguish between reliable and unreliable results.

As in the previous section, let us again assume that the flow field is corrupted by an additive noise (N_t, N_l). In addition to the properties and definitions already stated in Section 5, we assume that N_t and N_l are uncorrelated. Using the directional confidence measure, the depth estimate is given by Eq. (14). Assuming that the motion parameters are accurately recovered, the variance of each depth estimate is

$$\sigma^2(\tilde{Z}) = \frac{W_t^2 \sigma_t^2 {\alpha_U'}^2 + W_l^2 \sigma_l^2 {\beta_U'}^2}{\left(W_t {\alpha_U'}^2 + W_l {\beta_U'}^2 \right)^2}. \tag{26}$$

Substituting W_t and W_l with $1/\sigma_t^2$ and $1/\sigma_l^2$, as proposed in the previous section, we obtain:

$$\sigma^2(\tilde{Z}) = \frac{1}{\left(\alpha_U'/\sigma_t \right)^2 + \left(\beta_U'/\sigma_l \right)^2} \tag{27}$$

We can now define a confidence measure for $\tilde{Z}$:

$$C(\tilde{Z}) \stackrel{\text{def}}{=} 1/\sigma^2(\tilde{Z}) = W_t {\alpha_U'}^2 + W_l {\beta_U'}^2. \tag{28}$$

Notice that this confidence measure is small near the FOE, where α_U' and β_U' are close to 0.

Eventually, we are usually interested in estimating Z/r, that is, $1/\tilde{Z}$. Denoting Z/r by Z^*, we can obtain the following equalities, where estimated values are denoted by small letters:

$$\Delta Z^* \stackrel{\text{def}}{=} z^* - Z^* = 1/\tilde{z} - 1/\tilde{Z} = \frac{\tilde{Z} - \tilde{z}}{\tilde{Z}\tilde{z}} = \frac{-\Delta\tilde{Z}}{\tilde{z}\tilde{z}}. \tag{29}$$

Thus, the relative error in the estimated depth is

$$|\Delta Z^*|/Z^* = |\Delta \tilde{Z}| \quad \tilde{z} \quad \approx \quad |\Delta \dot{z}| \quad \dot{z} \quad = \quad |\Delta \tilde{Z}| \frac{Z}{r}, \tag{30}$$

where the approximation above is justified if the relative error in estimating $\tilde{Z}$ is small. In this case,

$$\sigma^2(\Delta Z^*/Z^*) \approx \frac{Z^2/r^2}{(\alpha'_U/\sigma_t)^2 + (\beta'_U/\sigma_l)^2} = \frac{1}{(\alpha'_T/\sigma_t)^2 + (\beta'_T/\sigma_l)^2}, \tag{31}$$

and a confidence measure can be defined as

$$C(\Delta Z^*/Z^*) = \frac{r^2}{Z^2}(W_t{\alpha'_U}^2 + W_l{\beta'_U}^2) = W_t{\alpha'_T}^2 + W_l{\beta'_T}^2. \tag{32}$$

As a conclusion, the estimated value of Z/r becomes more reliable as the ratio between the translation magnitude and Z is increased. In addition, notice that the reliability is determined by the ratios, σ_t/α'_T and σ_l/β'_T, between the expected measurement errors and the corresponding translational components. A reliable depth estimate can be expected only if at least one of these ratios is small.

7. EXPERIMENTS

In this section we compare results achieved by employing either a scalar confidence measure or a directional confidence measure. The first experiment is based on simulated data, while the other two are based on images taken from a video camera translating through a hallway in the direction of the line of sight.

7.1 Experiment 1

The first experiment simulates a camera translating along the line of sight at speed

of one (focal) unit per second. This motion can be represented by $\underline{T} = (0.,0.,1.)$ and $\underline{\Omega} = (0.^\circ, 0.^\circ, 0.^\circ)$. The environment consists of two planar surfaces parallel to the image plane. A background plane is in a distance of 20 units from the image plane. It is occluded around its intersection with the line of sight by the second surface, which is a planar patch in a distance of 10 units from the image plane. The field of view of the camera is 30°, and the image contains 512×512 pixels.

Velocity vectors are uniformly sampled in the image. Each vector is perturbed by additive noise, with two orthogonal and independent components, N_t and N_l. These noise components are assumed to be uniformly distributed in the intervals $[-0.5, 0.5]$ and $[-6., 6.]$, respectively, where values are given in units of pixels per second. The angle ρ, between the x-axis and the axis corresponding to N_t, is uniformly distributed in the interval $[0.^\circ, 180.^\circ)$.

The motion and depth values were computed from the flow data using both the scheme with scalar confidence measure and the scheme with directional confidence measure. In the first case, the confidence values should be identical for each velocity vector. In the second case, following the discussion in Section 5, $W_t/W_l = (6/0.5)^2 = 144$.

A statistical study of the results was performed, based on $\sim$ J experiments with each of the schemes. In 100 experiments 64 velocity vectors were used, while in the other experiments 256 vectors were used. In each experiment the noise values were randomly sampled. The results, shown in Table 1, demonstrate the significant improvement achieved by using the directional confidence measure. Relative to the scheme based on scalar confidence measure, there is an improvement of more than 50% in estimating the motion parameters, and more than 60% in estimating the normalized depth values. Notice that

100

a similar improvement in the estimation of the motion parameters (but not in the depth values) has been achieved by using 256 flow vectors instead of 64 vectors.

Errors	# Vectors	Scalar Confidence	Directional Confidence
ΔU	64	2.20°	1.05°
	256	0.88°	0.32°
$|\Omega|$	64	0.158°	0.078°
	256	0.078°	0.030°
$|\Delta \dot{Z}|/\dot{Z}$	64	23.4%	9.5%
	256	18.7%	6.3%

Table 1: Experiment 1. The average errors in the direction of the translation vector and in the magnitude of rotation, and the average relative error in the depth values.

7.2 Experiment 2

This experiment is based on a dense displacement field and a related directional confidence measure computed by Anandan's technique [Ana88]. The experiment demonstrates the ability to recover 3-D motion and structure from such estimates of image motion, using either a scalar confidence measure or a directional confidence measure. The input images (of 256×256 pixels), the displacement field and the maximal component of the directional confidence measure are shown in Fig. 6, Fig. 7 and Fig. 8, respectively. The field of view of the camera is 25°.

The confidence values, computed by Anandan's technique, can take any non-negative number. We assumed, however, that the standard deviation of the least accurate displacement measurements is at most 10 times the standard deviation of the most accurate measurements. Therefore, we transformed the confidence values W into the interval
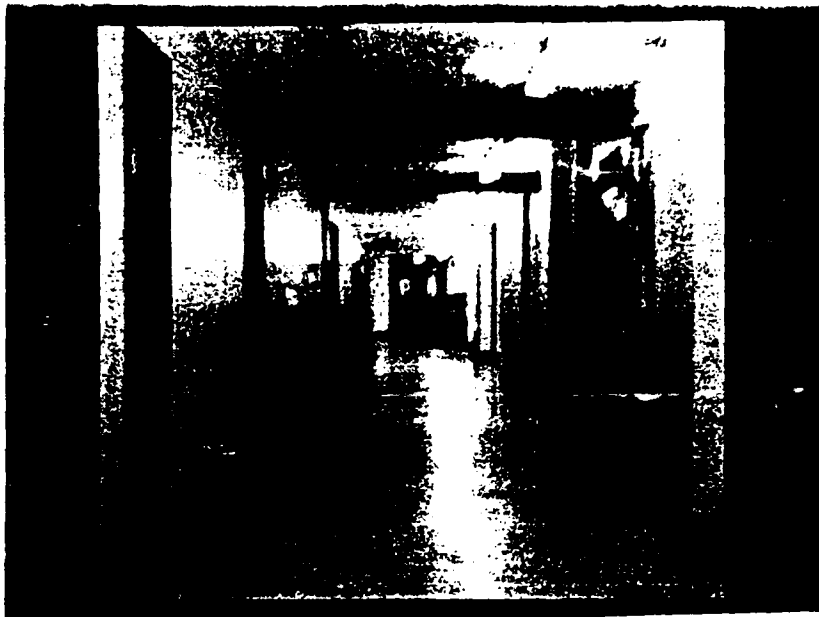
(a)

(b)

Fig. 6: Experiment 2: the intensity images.
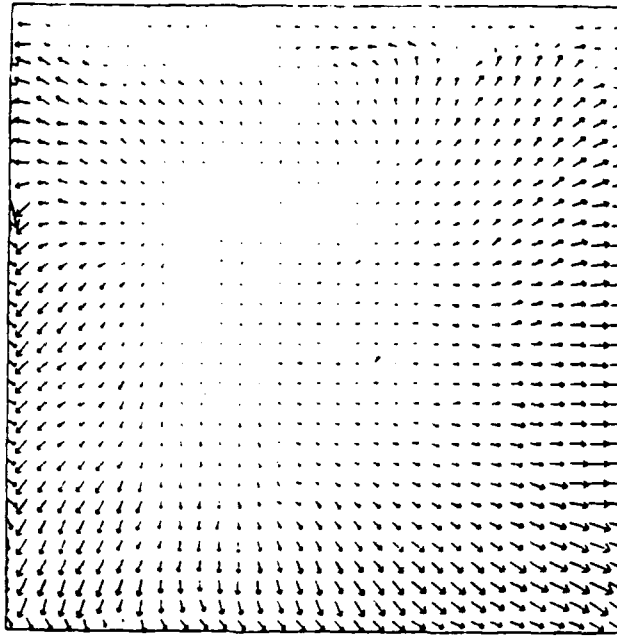
Fig. 7: Experiment 2: a 32×32 sample of the computed flow field. The vectors are scaled by 1.2.
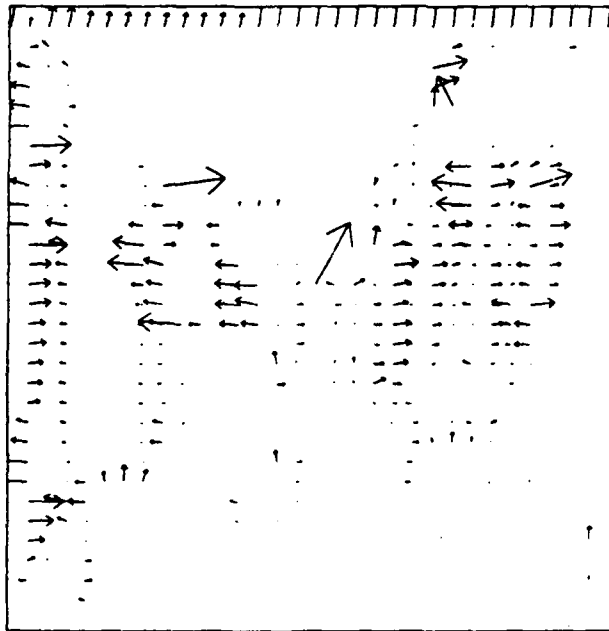
Fig. 8: Experiment 2: a 32×32 sample of the maximal component of the directional confidence measure. Notice the high confidence assigned to the normal component of displacements near straight lines.

[1., 100.), using the transformation $W' = (100W + 1)/(W + 1)$. Then, a scalar confidence measure was derived from the directional confidence measure using the relation (25). A selection of 256 vectors from the displacement field was performed, based on two criteria: high values of the scalar confidence measure, and a uniform distribution over the image.

The 3-D motion parameters were computed from the selected vectors by minimizing either Eq. (11), using the scalar confidence measure, or Eq. (15), using the directional confidence measure. In the first case we estimated the value of $\underline{U}$ as $(0.005, -0.020, 1.000)$, which is a deviation of $1.15°$ from the line of sight, and the value of $\underline{\Omega}$ as $(-0.035°, -0.112°, -0.652°)$. The results in the second case were almost identical.

In the last stage, the relative depth values were computed using either Eq. (10) or Eq. (14). In both cases (see Figs. 9 and 10) the depth values usually vary smoothly, unfortunately even across occlusion boundaries, due to the smoothness process in Anandan's technique. The results obtained by using the directional confidence measure seem to be somewhat better in this sense. The overall improvement is not significant however, because the tangential components of displacement vectors at edge points are almost as accurate as the normal components. This was achieved by employing the directional confidence measure as a tool in the smoothness process. To conclude, the directional confidence measure did not significantly improve the 3-D interpretation, since it did not reflect the accuracy of the displacement measurements.

7.3 Experiment 3 The input to this experiment is a list of line segment pairs, where the lines were extracted by the method described in [Bol87], and matched by the algorithm presented in [Wil88]. This experiment demonstrates the ability to recover 3-D motion and structure from line segments, using only two frames. As in the previous experiment, the
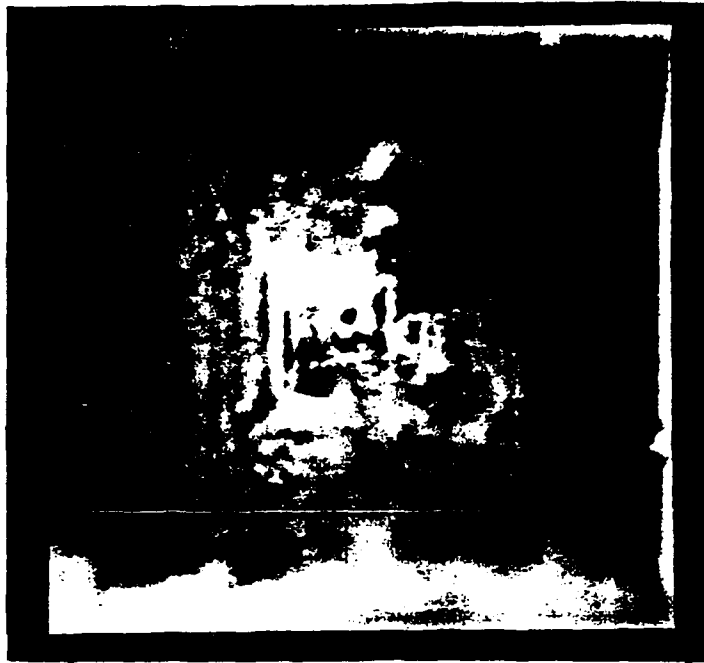
Fig. 9: Experiment 2: The depth map obtained by using the scalar confidence measure. The depth values are encoded by intensity (more distant surfaces are brighter).

Fig. 10: Experiment 2: The depth map obtained by using the directional confidence measure.

intensity images contain 256×256 pixels and correspond to field of view of $25°$. The first image is shown in Fig. 11, and the line segments computed for this image are shown in Fig. 12. The estimation of 3-D information was based on endpoint correspondences obtained from the list of line matches.

The endpoint pairs were grouped into two sets according to their reliability. This grouping affects the determination of the confidence measure, as explained in the following paragraph. In the set of unreliable correspondences, we included pairs associated with non-unique line matches or with matches where one segment was more then 20% longer than the other segment. We also included in this set pairs where one of the endpoints was less than 2.5 pixels away from the image boundary. All the other endpoint pairs were included in the set of reliable correspondences.

A directional confidence measure was determined for the displacement vector obtained from each endpoint pair. The direction of minimal confidence was estimated as the average orientation of the lines associated with the pair. For the reliable pairs, the standard deviations, σ_t and σ_l, of the transverse and longitudinal measurement errors, were estimated as 0.25 and 1 (in pixels), respectively. Hence, the corresponding confidence values were selected to be $W_t = 16$ and $W_l = 1$. For the unreliable pairs, we still selected $W_t = 16$, but W_l was determined to be 0. Thus, these pairs did not participate in the computation of the 3-D motion parameters, but their depth was estimated.

The 3-D motion and structure were computed using either a scalar confidence measure, or a directional confidence measure. In the first case, the same scalar weight was assigned to each of the more reliable endpoint pairs. The motion parameters found in this case were $\underline{U} = (0.045, 0.058, 0.997)$, corresponding to a deviation of $4.23°$ from the line of sight, and
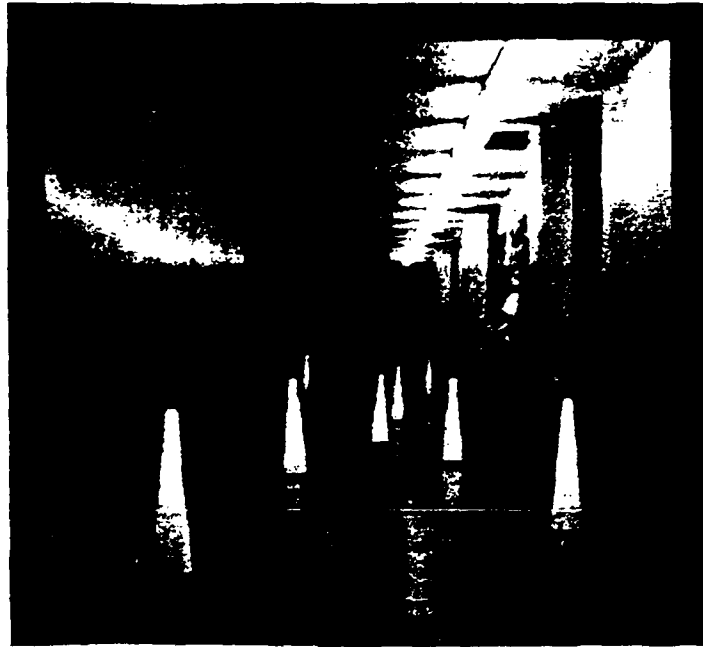
Fig. 11: Experiment 3: The first intensity image.
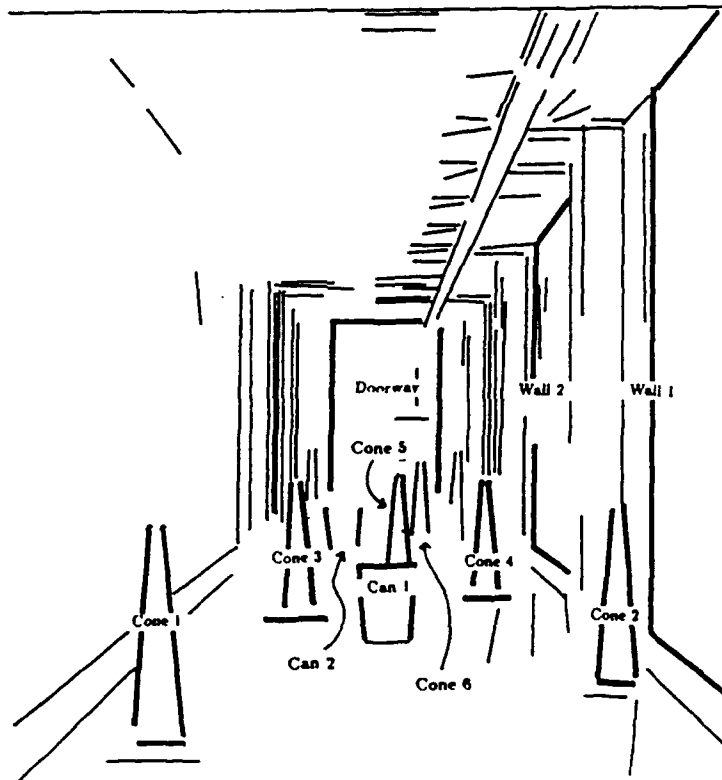
Fig. 12: Experiment 3: The line segments extracted from the first image. Objects with known depth values are labeled.

$\underline{\Omega} = (0.250°, 0.096°, -0.033°)$. In the second case, using a directional confidence measure, the vector $\underline{U}$ was found to be $(0.008, -0.011, 1.000)$, deviating $0.77°$ from the line of sight, and the rotation vector $\underline{\Omega}$ was estimated as $(0.087°, 0.204°, 0.041°)$.

In this experiment, the actual depth values of some of the objects in the scene are known. In addition, the translation magnitude is known to be 1.95 feet. In Table 2, we compare the estimated depth values computed by each of the algorithms to the ground truth values. For most objects, the estimates obtained by employing the directional confidence measure are significantly better than the estimates corresponding to the scalar confidence measure. The results for Cone 5 and Cone 6 are exceptional, because the related lines are oriented towards the FOE, and their longitudinal displacements are almost as accurate as the transverse displacements.

Object	Ground Truth	#Pairs	Scalar Conf. -- Average Error	Direct. Conf. -- Average Error	Direct. Conf. -- Aver. Norm. Err.
Cone 1	20.0 ft	8	11.5%	2.2%	0.88
Cone 2	25.0 ft	8	56.5%	1.8%	0.48
Wall 1	27.1 ft	4	7.1 %	5.6%	1.02
Can 1	30.0 ft	6	13.0%	6.8%	0.42
Cone 3	35.0 ft	6	14.6%	5.2%	0.45
Cone 4	40.0 ft	6	7.9%	6.2%	0.49
Cone 5	45.0 ft	4	13.0%	81.9%	1.01
Wall 2	48.7 ft	6	57.2%	39.8%	0.64
Can 2	55.0 ft	2	51.1%	55.5%	0.49
Cone 6	60.0 ft	4	32.8%	67.9%	1.09
Doorway	87.1 ft	8	91.3%	57.4%	0.84

Table 2: Experiment 3. For each object the following data is shown: the ground truth value, the number of endpoint pairs, the average value of errors $|\Delta \hat{Z}|/\hat{Z}$, both for the scalar confidence measure and for the directional confidence measure, and the average value of $|\Delta \hat{Z}|/\sigma(\hat{Z})$ for the directional measure.

The depth errors associated with the directional confidence measure were normalized to units of their estimated standard deviations. In other words, the ratios $|\Delta \bar{Z}|/\sigma(\bar{Z})$ were computed, using Eq. (27) for estimating $\sigma(\bar{Z})$. For each object, the average of these normalized errors was computed (see Table 2). These average values vary between 0.42 and 1.09, thus, demonstrating the predictability of the actual depth errors from their estimated standard deviations. This shows that Eq. (27) can be used successfully to distinguish between reliable and unreliable estimates.

8. SUMMARY

The directional confidence measure is a numerical representation of the expected reliability of image flow estimates. A scheme for incorporating this measure into a least squares technique for computing 3-D motion and structure from a flow field was introduced. A confidence measure for the depth estimates was also presented, and relations between these measures and between expected errors of the flow estimates were established.

The ability to employ a directional confidence measure was found to be especially useful in the case of line segment correspondences. Experimental results demonstrated the superiority of this measure over a scalar confidence measure in cases where the reliability of the image flow is orientation dependent and can reasonably be estimated.

ACKNOWLEDGEMENTS

REFERENCES

[Adi85a] G. Adiv, "Determining 3-D Motion and Structure from Optical Flow Generated by Several Moving Objects", *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, pp. 384–401, July 1985.

[Adi85b] G. Adiv, "Interpreting Optical Flow", Ph.D Dissertation, Computer and Information Science Dept., Univ. of Mass., 1985.

[Adi89] G. Adiv, "Inherent Ambiguities in Recovering 3-D Motion and Structure from a Noisy Flow Field", to appear in *IEEE Trans. Pattern Anal. Machine Intell.*, 1989.

[Ana84] P. Anandan, "Computing Dense Displacement Fields with Confidence Measures in Scenes Containing Occlusion", in *Proc. DARPA Image Understanding Workshop*, New Orleans, Louisiana, 1984, pp. 236–246.

[Ana87] P. Anandan, "Measuring Visual Motion from Image Sequences", Ph.D. Dissertation, Computer and Information Science Dept., Univ. of Mass., 1987.

[Ana88] P. Anandan, "A Computational Framework and an Algorithm for the Measurement of Visual Motion", to appear in *International Journal of Computer Vision*, 1988.

[Bol87] M. Boldt and R. Weiss, "Token-Based Extraction of Straight Lines", TR 87-104, Computer and Information Science Dept., Univ. of Mass., Amherst, Mass., October 1987.

[Bru81] A.R. Bruss and B.K.P. Horn, "Passive Navigation", MIT A.I. Memo 662, 1981.

[Dut88] R. Dutta, R. Manmatha, E.M. Riseman and M.A. Snyder, "Issues in Extracting Motion Parameters and Depth from Approximate Translational Motion", in *Proc. DARPA Image Understanding Workshop*, Cambridge, MA, 1988, pp. 945–960.

[Fau87] O.D. Faugeras, F. Lustman and G. Toscani, "Motion and Structure from Motion from Point and Line Matches", in *Proc. 1st Int. Conf. Computer Vision*, London, 1987, pp. 25–34.

[Liu88] Y. Liu and T.S. Huang, "Estimation of Rigid Body Motion Using Line Correspondences", *Computer Vision, Graphics and Image Processing*, vol. 43, pp. 37–52, 1988.

[Lon81] H.C. Longuet-Higgins, "A Computer Algorithm for Reconstructing a Scene from Two Projections", *Nature*, vol. 293, pp. 133–135, Sep. 1981.

[Med85] G. Medioni and Y. Yasumoto, "Robust Estimation of 3-D Motion Parameters

from a Sequence of Image Frames Using Regularization", in *Proc. DARPA Image Understanding Workshop*, Miami Beach, Florida, 1985, pp. 117–128.

[Nag86] H. H. Nagel and W. Enkelman, "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences", *IEEE Trans. Pattern Anal. Machine Intell.* Vol. PAMI-8, pp. 565-593, 1986.

[Spe87] M.E. Spetsakis and J. Aloimonos, "Closed Form Solution to the Structure from Motion Problem from Line Correspondences", in *Proc. Sixth National Conf. Artif. Intell.*, Seattle, WA, 1987, pp. 738–743.

[Tsa84] R.Y. Tsai and T.S. Huang, "Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces", *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 13-27, Jan. 1984.

[Ull79] S. Ullman, "The Interpretation of Visual Motion". Cambridge, MA: MIT Press, 1979.

[Wel87] W.M. Wells III, "Visual Estimation of 3-D Line Segments From Motion — A Mobile Robot Vision System", in *Proc. Sixth National Conf. Artif. Intell.*, Seattle, WA, 1987, pp. 772–776.

[Wil88] L.R. Williams and A.R. Hanson, "Translating Optical Flow into Token Matches", in *Proc. of DARPA Image Understanding Workshop*, Cambridge, Mass., 1988, pp. 970–980.

Towards Automatic Generation of
Object Recognition Programs

Katsushi Ikeuchi and Takeo Kanade
May 1988
CMU-CS-88-138

The views and conslusions 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 Projets Agency or the U.S. Government.

112

Abstract

This paper discusses issues and techniques to automatically compile object and sensor models into a visual recognition strategy for recognizing and locating an object in three-dimensional space from visual data. Historically, and even today, most successful model-based vision programs are handwritten; relevant knowledge of objects for recognition is extracted from examples of the object, tailored for the particular environment, and coded into the program by the implementors. If this is done properly, the resulting program is effective and efficient, but it requires long development time and many vision experts.

Automatic generation of recognition programs by compilation attempts to automate this process. In particular, it extracts from the object and sensor models those features that are useful for recognition, and the control sequence which must be applied to deal with possible variations of the object appearances. The key components in automatic generation are: object modeling, sensor modeling, prediction of appearances, strategy generation, and program generation.

An object model describes geometric and photometric properties of an object to be recognized. A sensor model specifies the sensor characteristics in predicting object appearances and variations of feature values. The appearances can be systematically grouped into aspects, where aspects are topologically equivalent classes with respect to the object features "visible" to the sensor. Once aspects are obtained, a recognition strategy is generated in the form of an interpretation tree from the aspects and their predicted feature values. An interpretation tree consists of two parts: a part which classifies an unknown region into one of the aspects, and a part which determines its precise attitude (position and orientation) within the classified aspect. Finally, the strategy is converted into a executable program by using object-oriented programming. One major emphasis of this paper is that sensors, as well as objects, must be explicitly modeled in order to achieve the goal of automatic generation of reliable and efficient recognition programs.

Actual creation of interpretation trees for two toy objects and their execution for recognition from a bin of parts are demonstrated.

Table of Contents

1. Introduction

A large class of practical vision problems is object recognition, that is, recognizing and locating objects in the scene by means of visual inputs. To name a few, visual part acquisition on a conveyer belt or from a bin of parts, target recognition in aerial images, and landmark recognition by a mobile robot, all belong to this class of problems. In most of these cases, we have some prior knowledge of the objects of interest, such as the shapes, sizes, reflective properties, and so forth. Model-based vision [7, 18] seeks to actively use such prior knowledge of objects for guiding the recognition process in order to achieve efficiency and reliability.

One of the critical issues in building a model-based vision system is how to quickly extract and organize the relevant knowledge of an object and to systematically turn it into a vision program. In particular, it is important to know what features of objects are useful for recognition, and what control is to be applied to deal with possible variations of the object appearances. In earlier vision systems, such knowledge of objects has been extracted from examples of the object, tailored for the particular environment, and coded into the program by the implementor. For example, in interpreting incomplete line drawings of polyhedra of known size and shape, Falk [20] analyzed failure patterns of line extraction and implemented strategies to cope with them. In fact, even today, most successful vision systems are developed based on the implementors' insight into the specific problems. Some representative examples include 3D object recognition systems in range maps by Oshima and Shirai [53] and by Faugeras and Hebert [21], aerial photointerpretation systems by Nagao and Matsuyama [51] and by McKeown, Harvey and McDermott [47], bin-picking systems by Perkins [56] and Ikeuchi and Horn [35], and the NAVLAV mobile robot vision system by Thorpe. et al [61]. In these systems, features and recognition strategies to be used are selected by the researchers. Although the resulting system may be effective and efficient, this "hand-coding" method requires large amounts of time and deep vision expertise for building model-based vision systems.

Quite often, a geometrical model of the object is available which represents the three-dimensional shape information by means of polyhedra, generalized cylinders, or other primitives. Given such an object model, visual recognition of an object amounts to determining its attitude (position and orientation) in space by using its various features which are observable in the images. In this view, one can imagine a generic model-based vision system which, given an input image or other sensory data, recognizes an object in it by means of a geometric

reasoning mechanism which can deduce possible object attitudes from apparent object features. The historical and pioneering vision system by Roberts [58] can be viewed as such a generic approach. It reduced the problem of object matching to that of estimating the parameters of transformation (rotation, translation, size, and projection) by minimizing a matching error between model vertices and image joints.

Grimson and Lozano-Perez [25] have formulated the problem of object localization measurements (such as position) within a hypothesize-and-test search paradigm. When matching a set of observed surface points with a set of polyhedral object models, the possible matching pairs are expanded as a search tree. The matcher prunes this tree by using relational constraints between pairs of measurements which the object models impose if the matching is correct so far. The method has been applied to 2D and 3D object recognition using sparse range, touch, and orientation sensory inputs.

Probably, however, the most representative effort toward domain-independent model-based vision systems is ACRONYM by Brooks [12]. ACRONYM takes models of objects represented by generalized cylinders and their spatial relationships. Recognition or matching of the models to an input image is performed by using a symbolic algebraic reasoning system which reasons about projection and relational constraints on geometry. ACRONYM has succeeded in recognizing airplanes in aerial images.

When performing matching, a generic domain-independent model-based system relies on a generic reasoning mechanism: numerical optimization of some matching criterion, constraint satisfaction by symbolic reasoning, or tree search by hypothesize-and-test. As a result, the system uses the object model interpretively, that is, the knowledge is extracted from the model and transformed into an execution strategy at run time. As a result, the system may not be most efficient for the particular object in hand. This is a necessary price that an interpretive method must pay for its generality and flexibility.

One method for increasing efficiency is compilation. That is, the relevant knowledge in the object models is extracted and compiled into an object recognition strategy off-line so that as little computation as possible is spent at run time. Interestingly enough, we can regard some of the earlier vision work as examples of compilation. The generalized Hough transform by Ballard [3] and the direction coding method by Yoda, Motoike, and Ejiri [65] can be regarded as

116

compiling the object shape in the appropriate transform so that the recognition reduces to peak finding in a histogram. However, these methods have limited applicability.

Bolles and his colleques used a "local-feature-focus" recognition strategy for recognition of 3D objects in a jumble [8, 9]. The method involves selecting a class of "focus" features of similar shape on the object. Matching begins with the "focus" features. In selecting appropriate features for the strategy, they precomputed various feature values from a given CAD model of objects.

. Goad [23] presented one of the first and most systematic methods for automatic generation of object recognition programs based on compilation. His method compiles visibles edge of an object into an interpretation tree. Each branch of the tree is constructed to execute three stages: prediction, observation, and back-projection. In the prediction stage, a model edge is extracted from the node based on the current hypothesis of viewer direction, and the position and orientation of its projection in the image is predicted. In the observation stage, the list of image edges is checked to see whether any has the predicted qualities. In the back-projection stage, if an edge with predicted qualities was found in the prediction stage, then the match is extended to include this edge, and the measured position and orientation of the edge are used to refine the current hypothesis as to the location of the camera. During the compilation mode, stages and nodes which will become unnecessary at run time are detected and pruned. Various conditions and data structures to be used at run time are also computed. This way, much of the computation at run time is saved. The method for selecting the most efficient sequence of edges to be examined was not discussed, however.

Koezuka and Kanade [41] constructed an interpretation tree automatically from a model of a polyhedral object by using parallel edges as initial features to be used in matching. Parallel line features remains parallel over a wide range of viewing directions, but the direction and distance between a pair of lines still provide strong constraints on viewer direction, and can be used to create a reliable and efficient interpretation tree.

Ikeuchi [34] presented a compilation technique based on visible regions. The system classifies various views into aspects, where aspects are defined as topologically equivalent views. The interpretation tree is constructed so that an unknown view will be classified into an aspect and then its attitude will be determined precisely. He developed rules to generate an interpretation tree from a geometric model. The rules determine what kinds of features should be used in what

117

order and generate an interpretation tree.

Automatic generation of recognition programs by compilation of object models tries to combine the merits of a hand-written system and those of a generic interpretive system. A general compilation program generates a tailored special program from a given 3D model. A large portion of the computation needed for using the object model, such as analysis of the best recognition strategy, analysis of occlusion, and estimation of expected feature values, can be done at compile time, and the result can be compiled into the special program. In some cases, the object properties might be represented in the flow of the program rather than its data structure. As a result, the compiled special program to run on-line can be more efficient than generic programs. Yet, since the program is generated automatically, the development time could be reduced.

This paper discusses issues and techniques for automatic generation of recognition programs by compilation. The discussion will be based on our current approach, whose key steps are object modeling, sensor modeling, prediction of object appearances, strategy generation, and program generation. An object model describes geometric and photometric properties of an object to be recognized. A sensor model specifies the sensor characteristics in predicting object appearances and variations of feature values. The appearances can be systematically predicted and grouped into aspects, and a recognition strategy is generated in the form of an interpretation tree from the grouping and the predicted feature values. Finally, the strategy is converted into an executable program by using object-oriented programming. A major emphasis of this paper is that sensors, as well as objects, must be modeled explicitly in order to achieve the goal of automatic generation of reliable and efficient recognition programs. First, we will present our initial system for generating an interpretation tree for bin-picking using photometric stereo. This example system will introduce various concepts as well as issues.

2. Compiling an Object Model into an Interpretation Tree

This section will present an example of compilation of a geometric object model into an interpretation tree. The example task is a bin picking task. The object shown in Figures 1 (a) and (b) is the sample object and the scene in Figure 1 (c) is a typical image from which the object must be recognized and located.
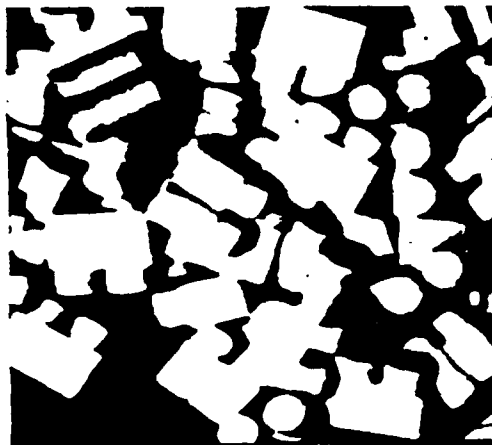
A 3D object can give rise to an infinite number of 2D shapes in an image. These apparent 2D
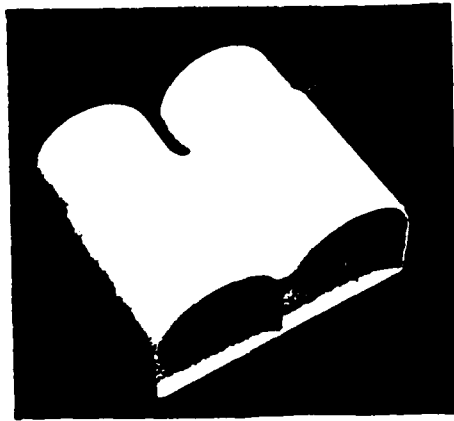
(a)
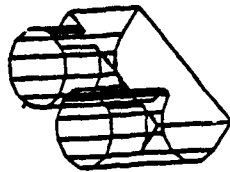
(b)

(c)

Figure 1: Object recognition example: (a) Photo of a sample object; (b) Geometric model of the object; (c) Sample scene.

shapes of a 3D object, however, can be grouped into a finite number of equivalence classes, called *aspects* [39, 40], where each aspect contains the apparent shapes arising from the same set of visible features of objects, such as faces, edges or vertices, with the same topological relationships among them. We can therefore distinguish two types of shape changes: one is shape change between aspects (called aspect change); the other is shape change within an aspect (called linear change). Figures 2(a) and 2(b) show examples of an aspect change and a linear change, respectively, for the object in Figure 1.

Use of aspects for object recognition has been proposed by many researchers. Our goal here is, given a model of an object, to automatically develop an interpretation tree which first classifies the input image of an object into one of the possible aspects, and then calculates the exact attitude of the object. It should be noted that different features are most likely required to resolve aspect changes than are required to resolve linear changes. Also, in resolving linear changes, appropriate techniques and features might be different depending on the particular aspect in which the linear change occurs. Thus, it is essential for both competence and efficiency to compile a geometrical model into an interpretation tree so that the most appropriate features among all the available features are used at each determination stage to resolve aspect and linear changes.

2.1. Extracting Aspects

For object recognition purposes, aspects are defined as topologically equivalent classes with respect to the object features "visible" to the sensors. For example, aspects have been defined by visible lines [40, 16]; by visible vertices [60, 62]; and by occluding boundaries [28, 26]. As will be explained later, our example system will use photometric stereo [64, 33] as the major sensor. Photometric stereo determines surface orientations by illuminating the surface with three light sources. Thus we categorize the aspects based on visible faces for photometric stereo.

Viewer or camera configurations, which result in various appearances of a 3D shape, consist of six degrees of freedom in general: three degrees of freedom in translation, and three degrees in rotation. However, in most industrial vision problems, such as bin picking, we can assume orthographic projection as the first approximation. This is because the camera is set up at a relatively far and fixed distance to the objects and the objects are imaged only near the center of the camera's field of view. This means that the three translations are either known or constant
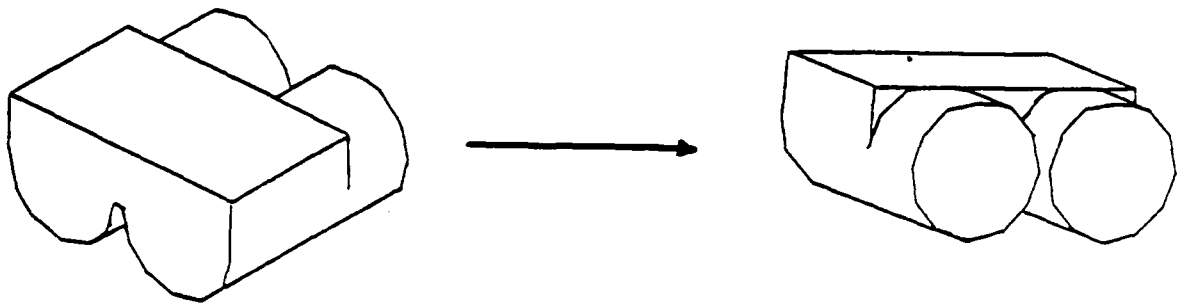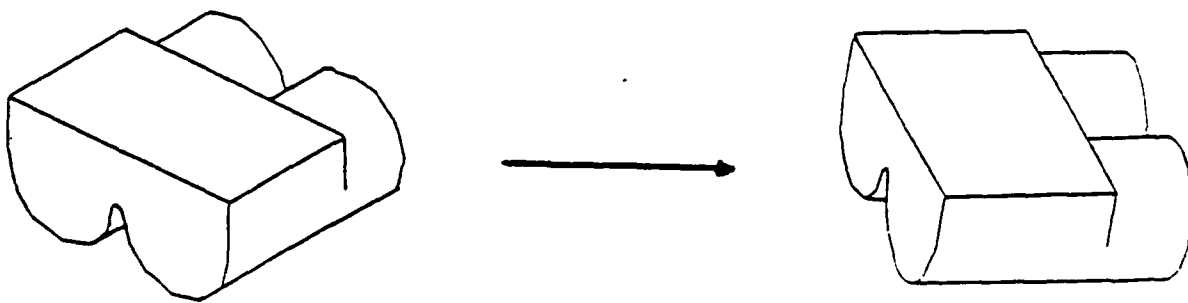
Figure 2: Examples of aspect change and linear change of object appearances: (a) Aspect change where sets of visible surfaces differ; (b) Linear change where only the shape of each surface is skewed.

Since a rotation around the camera optical axis results in a rotation of the image, not in a change of appearances, the two degrees of freedom which specify the viewer direction are the dominant ones in determining aspects.(See Figure 3).

We will thus explore changes of apparent shapes over the set of possible viewer directions. A viewer direction has two degrees of freedom and can be described as a point of the Gaussian sphere which is placed at the center of an object.

Each apparent shape (thus, each point on the Gaussian sphere) can be characterized by those faces visible from that viewer direction. Suppose we have n faces, $S_1, S_2, ..., S_n$, where one face corresponds to either a planar surface or a curved surface which will be detected as a single surface patch in photometric stereo. Let the variable X_i denotes the visibility of face S_i, that is

$$X_i = \begin{cases} 1 & \text{face } S_i \text{ is visible;} \\ 0 & \text{otherwise.} \end{cases}$$

An n-tuple $(X_1, X_2, ..., X_n)$ represents a label of an apparent shape in terms of face visibility. This label will be referred to as a *shape label*, and we can characterize each viewer direction with this label.

The set of contiguous viewer directions that have the same *shape label* forms an *aspect*. There are two methods to enumerate possible aspects of a given object: an analytic method and an exhaustive method. Though precisely finding possible aspects by an analytic method is relatively easy for convex polyhedra, it becomes more complex and less tractable for concave objects and curved objects. For practical purposes, we favor the exhaustive method, in which we generate apparent shapes of the object under various viewer directions sampled on the Gaussian sphere, examine shape labels of the generated shapes, and classify them into aspects.

We tessellate the Gaussian sphere by using a geodesic dome which subdivides the sphere into many small spherical triangles [14], each of which represents a sampled viewer direction. These sampled viewer directions evenly cover the whole surface of the Gaussian sphere surface. At each sampled viewer direction, an apparent shape of the object is generated using a geometric modeler, and its shape label $(X_1, X_2, ..., X_n)$ is calculated. This way, all possible shape labels are
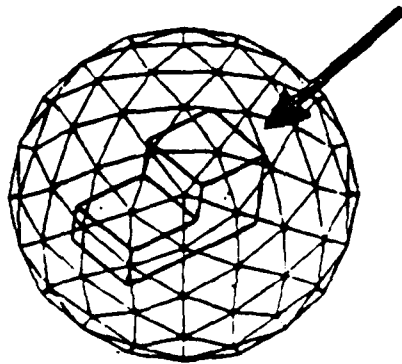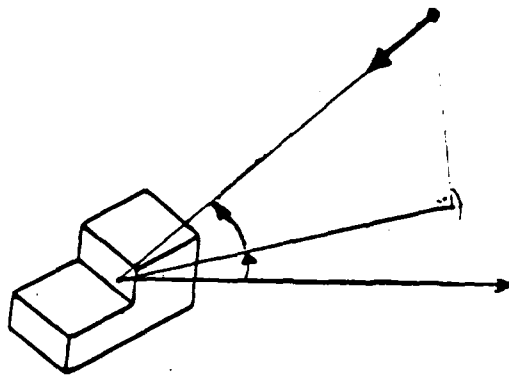
VIEWER DIRECTION

Figure 3: Two degrees of freedom in viewer directions and their representation in the tessellated Gaussian sphere.

123

calculated, evenly sampled over all possible viewer directions, and grouped into aspects[1]. Finally, a representative attitude is selected for each aspect chosen from the set of viewer directions which result in the same aspect. Usually, the viewer direction which results in an appearance with the largest sectional area is selected as the representative attitude. The viewer rotation for the representative attitude is determined so that the direction of maximum moment of the appearance agrees with the x axis of the image plane. The representative attitude is used to calculate the representative values of features to be used to discriminate aspects and to calculate the precise attitude within an aspect.

Figure 4 shows the result of applying this method to the object of Figure 1. The sample object has twelve component faces. Figure 4(a) shows the geometric model of the object. Figure 4(b) shows the Gaussian sphere tessellated into sixty small triangles using the one-frequency dodecahedron. Sixty different shapes corresponding to the tessellated triangles are generated as shown in Figure 4(c), where the faces surrounded with bold lines are detectable using photometric stereo. Because of the geometry of the light sources, some faces visible to humans are not detectable by photometric stereo. Figure 4 (d) shows the larger eight component faces used for the shape label among the twelve faces of the object. Smaller regions under a certain threshold are regarded as non-detectable. Figure 4 (e) lists the five aspects obtained as the result of classification of the sixty appearances in Figure 4 (c). The visible faces are indicated under each aspect. For example, faces 1, 2, and 3 are observable in aspect 1, whose shape label is 11100000. For aspects 1 to 5, five representative attitudes are generated as shown in Figure 4 (f).

2.2. Sensors and Features

This section will give a brief description of the sensors we used and then present how the aspects are described in terms of available features. In our example system, the major sensor is photometric stereo which provides surface orientations. In addition, we use dual photometric stereo to obtain depth information and an edge detector to locate fine features of objects.

Photometric Stereo [64]

---

[1]Note on effectiveness and practicality of this method: constant cost; possible omission of aspects, but it would not hurt anyway because of its narrow visibility.
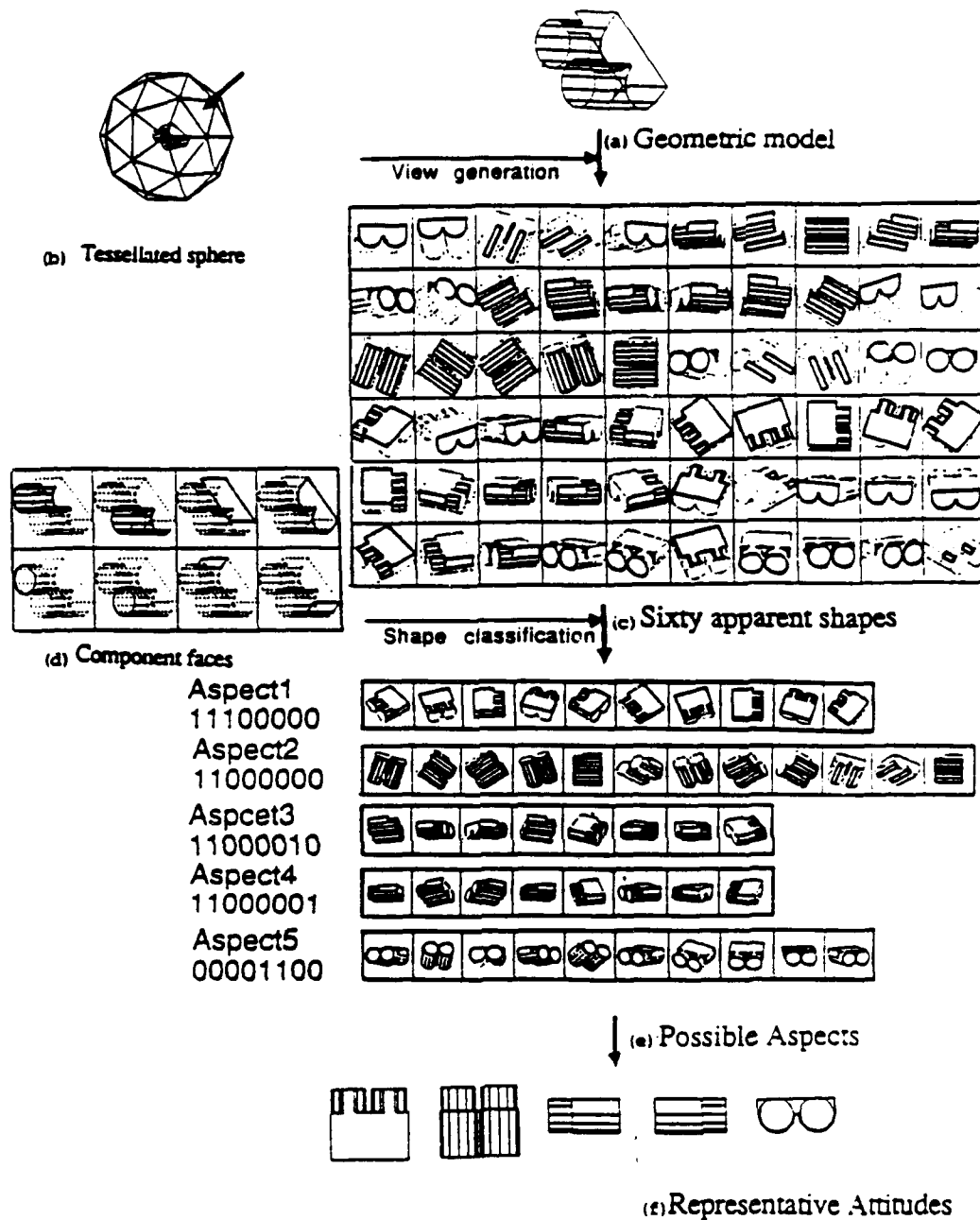
(a) Geometric model

View generation

(b) Tessellated sphere

(d) Component faces

Shape classification

(c) Sixty apparent shapes

Aspect1
11100000

Aspect2
11000000

Aspcet3
11000010

Aspect4
11000001

Aspect5
00001100

(e) Possible Aspects

(f) Representative Attitudes

Figure 4: Extraction of aspects: (a) Geometric model of an object; (b) Gaussian sphere tessellated into sixty triangles to represent viewer directions sampled; (c) Sixty appearances. The faces surrounded with bold lines are detectable by photometric stereo. Because of the geometry of the light sources, some faces visible to humans are not detectable by photometric stereo; (d) Eight component faces to be used for the shape label; (e) Five aspects obtained as the result of classification of sixty appearances by the shape label. Eight digits at each aspect represent the shape label of the aspect; (f) Five representative attitudes.

using these three sensors. In describing aspects, we can use features available from these three represetnations of the input scene. Since surface orientation is obtained as the needle map, we can actually recover 3D features of the original faces, instead of 2D projected features, such as the area, shape, etc. Let (p,q) be the surface gradient of a region. Then, the matrix

$$T = \begin{bmatrix} \sqrt{1+p^2} & pq/\sqrt{1+p^2} \\ 0 & \sqrt{1+p^2+q^2}/\sqrt{1+p^2} \end{bmatrix}$$

gives the affine transformation to map from the 2D image coordinates to the 2D coordinates on the 3D face. This transformation can be used to recover the 3D features of the original face from 2D features of the corresponding region in the image.

Each aspect is now described by using various features obtainable from the above sensors. In our example, features used include face moment, face relationships, face shape, edge relationships, extended Gaussian image (EGI), and surface characteristic distribution. Each of these features is discussed below.

Face moment

The face moments are represented by the two principal moments of inertia, m_{xx} and m_{yy}, of a face. These inertia moments roughly describe the shape of a face. More detailed shape information is represented by another feature.

Face relationship

An object often appears as multiple separated regions in the image. This is especially true with non-convex objects under photometric stereo. The relationships between regions are very useful features. For each visible face, relative position information is stored which tells where each of the other visible faces should appear in the aspect. The relationship is represented by a vector with respect to the local face coordinate system. The origin of the local coordinate system is the mass center. The z-axis and x-axis agrees with the surface orientation, the direction of the

Photometric stereo takes multiple images of the same scene from the same camera position under different illumination directions in order to determine surface orientations (p,q) based on differences in brightness. Since different images are taken from the same point, there is no disparity between the images as there is with binocular stereo, so no correspondence problem has to be solved. This makes photometric stereo very fast. By using photometric stereo we generate a needle map, which is a distribution of (p,q) over the image. From the distribution of (p,q) over a region, we can recover various geometric features of visible regions such as area and moment.

Dual Photometric Stereo [33]

Although photometric stereo can determine the surface orientation vary fast, it cannot determine absolute depth. In order to determine absolute depth fast, we exploit binocular stereo based on a pair of needle maps, each of which is obtained by photometric stereo.

A needle map obtained by photometric stereo can be easily segmented into isolated regions using uninterpretable regions around objects. Due to the arrangement of the light sources, a higher object projects shadows over the surrounding lower objects. Since the projected shadow areas becomes uninterpretable regions, a higher object is usually surrounded by uninterpretable regions.

We will establish the correspondence between left regions and right regions by using three characteristics: vertical mass center positions, average surface orientation over the region, and region area. Since our method only checks correspondences between regions, the number of combinations necessary to examine is small, so the system is very rapid. A depth map is obtained from each region's disparity and average surface orientation. The depth map will be used to determine the target region from which the recognition process begins.

Edge Detector

We also use an edge map which is   ined by differentiating brightness distributions with a Canny edge detector [15] and grouping edge points into line segments with a Miwa line finder [50]. The edge map will be used as a supplementary source when the system cannot determine the object attitude completely using features from a needle map.

In summary, an input scene is described by a needle map, a depth map, and an edge map by

maximum moment, respectively[2]

Face shape

The face shape is described by the radial distance function $d=d(\theta)$, where d is the radial distance from the mass center of the face to its boundary, and θ is the angle from the x-axis of the local coordinate system.

Edge relationships

In some cases the needle map cannot determine the object attitude uniquely. In this case some of the prominent edge information is useful to reduce ambiguity. The locations of edges are stored by the start and end positions. As in other face information, these positions are denoted in the local face coordinate system. When applying this information, a position is converted into a position on the image plane using the inverse affine transformation matrix derivable from the surface orientation of the face. Then, the narrow stripe region connecting the converted start and end positions can be searched on the edge map to see whether or not there is actually an edge.

Extended Gaussian image (EGI)

An EGI of an object is nothing but a spatial histogram of its surface orientations [31, 32, 13, 30, 44]. The EGI has two nice properties. One is that the EGI is invariant to translation of the object, and the other is that when an object rotates, its EGI also rotates in exactly the same manner while not changing the relative EGI mass distribution.

Surface characteristic distribution

A surface patch can be characterized as planar, cylindrical, elliptic, or hyperbolic. The characteristics are defined in terms of the Gaussian curvature and the mean curvature [11, 5] and are independent of the viewer direction and the rotation. Distribution of the characteristics are stored with respect to the local coordinate system, and are used in a similar way and for a similar purpose as prominent edges.

---

[2]This local coordinate has 180 degree ambiguity with respect to the x-axis direction. Also, if the region has no unique maximum moment direction, for example, a circular region, only the direction of x-axis is defined arbitrary. In this case, only the distance between the two region is stored

For each aspect extracted for an object, the features listed above are calculated. The descriptions of all aspects thus obtained are now used to construct the interpretation tree with which input scenes will be recognized.

2.3. Generating an Interpretation Tree

An interpretation tree consists of two parts: the first part is used for classifying the input scene into one of the aspects, and the second part is used for calculating the exact attitude of the object within the aspect determined. In this subsection we will create an interpretation tree for our example object. First we discuss how to generate the classification part of the interpretation tree. The basic idea is a recursive examination of features of aspects to see whether or not they can discriminate a group of aspects into sub-groups of aspects. That is, starting with all the aspects as a single group, we check if a certain feature can divide the group into subgroups. If so, a branch node is created which registers the feature as the discriminator and the subgroups divided are connected as descendant nodes. Then for each subgroup (descendant node), the process is applied recursively until a subgroup is made of a single aspect or equivalently a single aspect is assigned to a leaf node.[3]

We have used the following seven features for discrimination. In order of preference, they are: the original face moment, the original face shape, the extended Gaussian image (EGI), the surface characteristic distribution, the edge distribution, the region distribution, and the relationship between a particular edge and a particular surface characteristic distribution.

As an example, we apply this method to the object shown in Figure 1 (a). The object has five aspects, shown in Figure 4(e), so the start node contains a group of five aspects, {S1, S2, S3, S4, S5}[4] See Figure 5. Since the original face moment can divide the aspect groups into three sub-groups, {S1}, {S2, S3, S4}, and {S5}, it is adopted as a discriminator at the starting node, and three descendent nodes, N1, N2, and N3 are generated from the start node.

---

[3]Actually, as an initial stage of the project, a "skeleton" of a tree was predesigned by considering the "distances" among aspects, and the decision as to whether or not a feature can divide the aspects at the node was made by human. For more details, see [34]. This human-assisted decision process has since been converted to an automatic decision process.

[4]Moreprecisely, one aspect component, having the largest area, is selected among aspect components of each aspect as the face from which recognition process begins. Thus, the later stages examines various features of the selected aspect components.

Since node N1 and node N3 contain only one aspect, S1 and S5, respectively, the generation process terminates at these nodes. On the other hand, node N2 contains three aspects, so further processing is applied to the node. Neither the original face shape, the extended Gaussian image, the surface characteristic distribution, nor the edge distribution can not discriminate the aspect group {S2, S3, S4}. Since the region distribution divides the aspect group into two sub-groups, {S2} and {S3, S4}, this feature is adopted as a discriminator for node N2, and two descendent nodes, N21 and N22 are generated from N2. Node N22 still contains two aspects, and requires further processing. Because S3 and S4 have a different internal structure of regions, the region distribution feature is adopted as the discriminant to produce two nodes, N221 and N222. Now the complete aspect classification part of the interpretation tree has been obtained.

Once the aspect classification part is constructed, we will move on to generation of the part of the interpretation tree which determines the viewer direction and rotation. If a feature can reduce some of the remaining freedom in the viewer direction and rotation, it will be adopted into the tree. The decision as to whether or not a feature can reduce the freedom was made by a human at this point[5].

We have used the following eight features for determination of the linear shape change. In order of preference, they are: the mass center of EGI distribution, the EGI, the position of observable region distribution, the moment direction of original face, the original face shape, the position of the surface characteristics distribution, the position of the edges, and the position of the edges with respect to the position of the surface characteristics distribution.

The viewer direction and rotation are determined for each aspect using the most effective feature at each step. The selection depends on the aspect and the stage of the determining process. As an example, we will consider the case of node N21 or aspect S2. The other cases can be treated in the same way. Aspect S2 has two observable regions of cylindrical surfaces. The EGI mass center can determine viewer direction. Theoretically, the EGI distribution could have determined the viewer direction and the rotation uniquely in this aspect, but due to noise it would have been very unreliable. Thus, we will use other features to determine the viewer rotation.

Since aspect S2 has two observable regions, the region distribution feature is applicable and

---

[5]This human-assisted decision process has since been converted to an automatic decision process.

can constrain the viewer rotation up to two directions (up or down). Neither the moment direction, original face shape, nor surface characteristic feature can disambiguate one of the two remaining possibilities. However, the edge distribution feature can do. As a result, the EGI mass center, region distribution, and edge distribution have been adopted into the tree in this order. Figure 5 shows the final interpretation tree obtained.

2.4. Applying the Interpretation Tree

In recognizing objects at run time with the interpretation tree created, the system uses three kinds of feature maps: an edge map, a needle map, and a depth map as shown in Figure 6. An edge map is obtained by differentiating the camera intensity image. Each of two photometric stereo sensors, left and right, produce a needle map using three intensity images corresponding to the three lighting conditions. A depth map is constructed by the dual photometric stereo method [33], which matches a pair of needle maps, one from the left camera and one from the right camera. An important advantage of these three maps is that they are registered in the same coordinate system; that is, all pixels having the same $i-j$ pixel coordinates correspond to the same physical point.

Our bin-of-parts example scene contains many instances of the object, while the interpretation tree specifies how to recognize a single object. Therefore we have to select a portion of an image where the interpretation tree is going to be applied. For this purpose, we choose the highest region (ie, the region closest to the camera) as the target region to be interpreted.

The interpretation tree extracts necessary features from the region. These features will be transformed and compared with the aspect model according to the procedures contained in the interpretation tree. Based on the decisions at each node, the target region is classified into one of the aspects, and then the precise attitude and position are determined.

Figure 7 illustrates how the interpretation proceeds for the case of Aspect 2. The white arrow in the picture (b) indicates the target region. According to the interpretation tree, the face moment of the region is calculated by using the shape and size of the region together with its spatial surface orientations from the needle map. The rectangle in Figure 7 (a) indicates the direction and magnitude of the moment value thus obtained. Based on the value of face moment, the interpretation tree determines this region to belong to the group of aspects S2, S3, and S4.
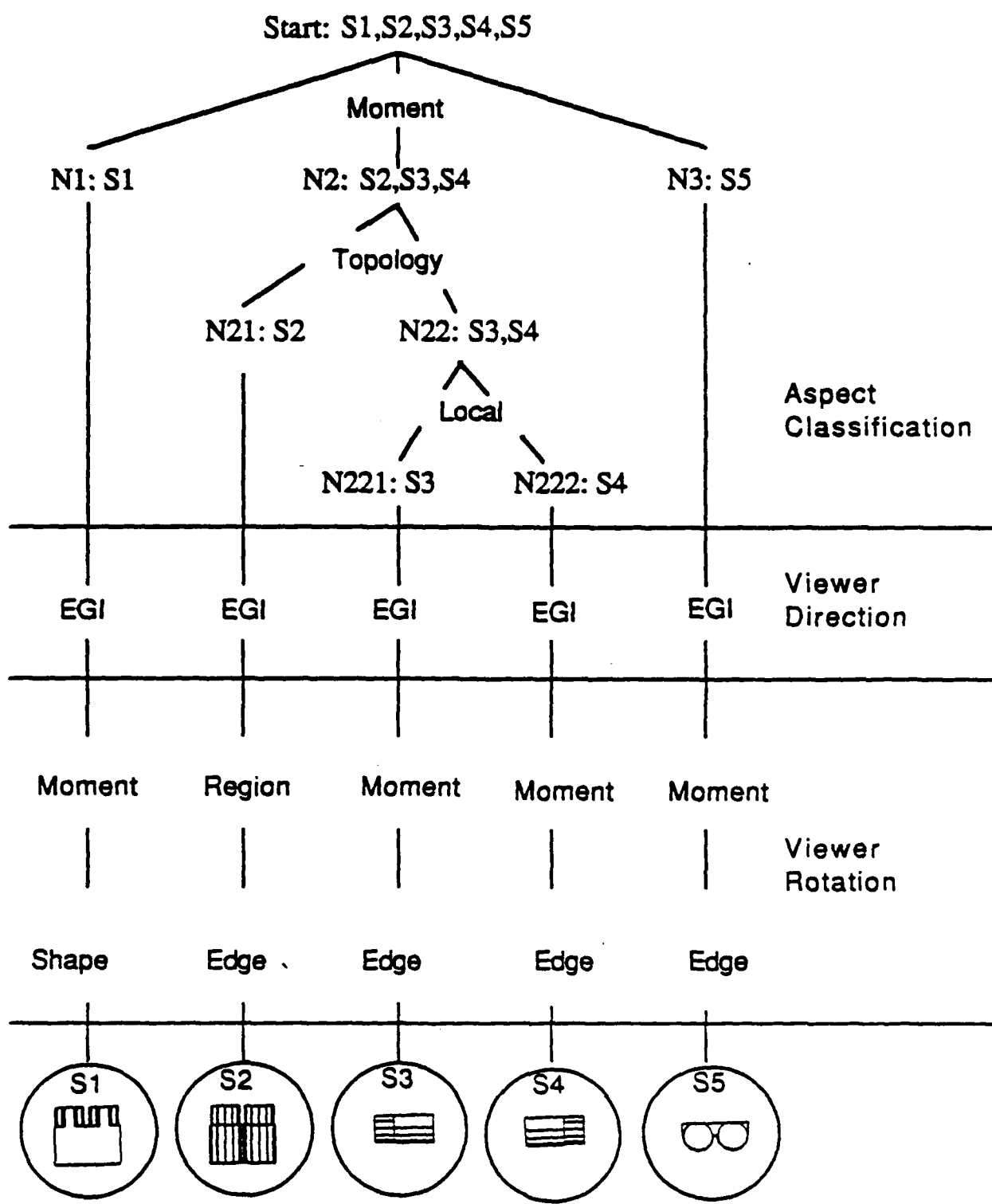
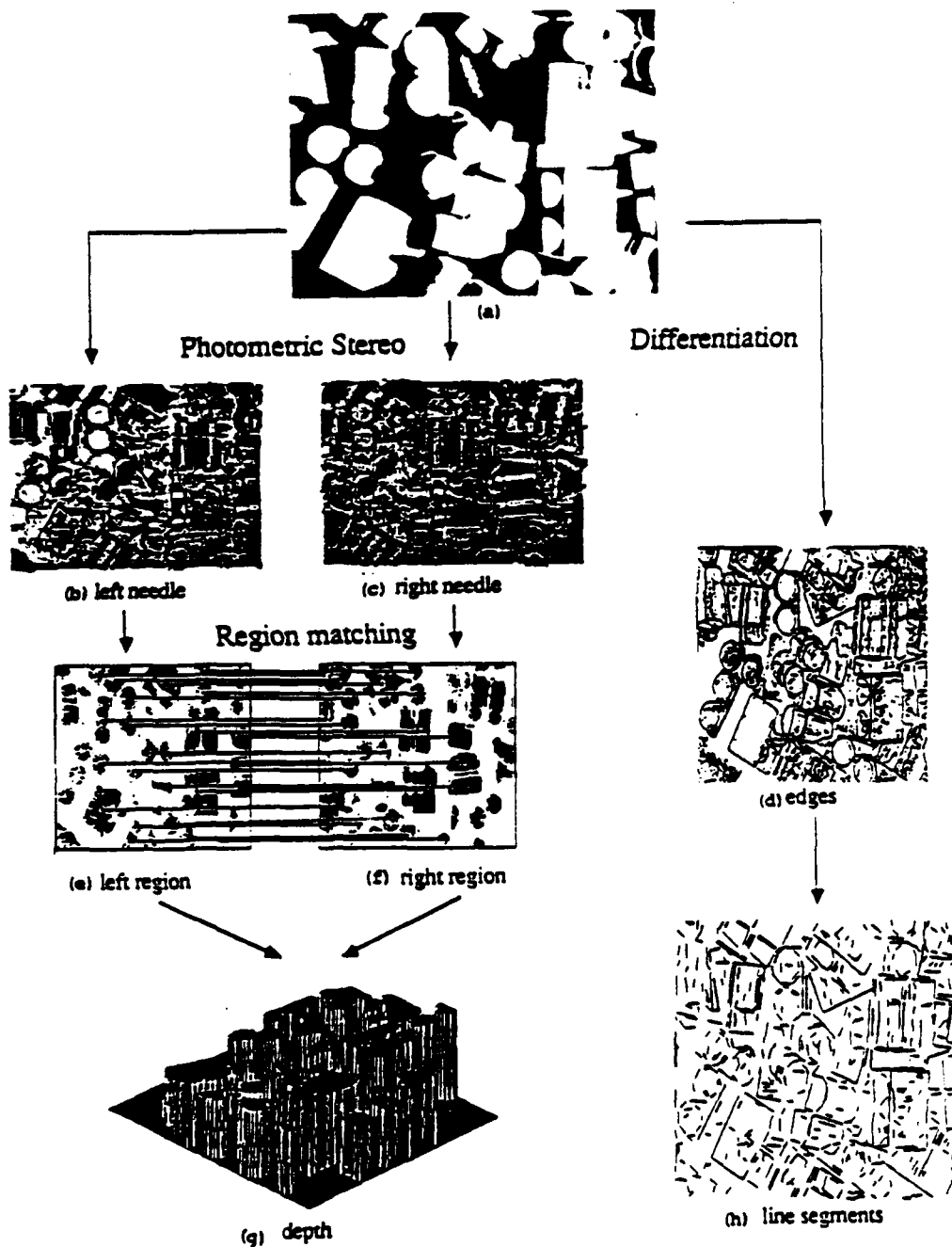**Figure 5:** Interpretation tree

130

Figure 6: Basic vision module: (a) Input scene; (b) Left needle map obtained by left photometric stereo. Surface orientations are depicted as small needles; (c) Right needle map obtained by right photometric stereo; (d) Edges obtained by Canny edge operator; (e) Left region map. A needle map obtained by photometric stereo can be easily segmented into isolated regions using uninterpretable regions around objects. Due to the arrangement of the light sources, a higher object projects shadows over the surrounding lower objects. Since the projected shadow areas becomes uninterpretable regions, a higher object is usually surrounded by uninterpretable regions. The left region map is obtained by segmenting the left needle map based on these uninterpretable regions; (f) Right region map; (g) Depth map obtained by dual photometric stereo. The correspondence between left regions and right regions is established by using three characteristics: vertical mass center position, average surface orientation over a region, region area. A depth map is obtained by fitting a plane based on the depth at a mass center given from disparity and average surface orientation; (h) line segments obtained by Miwa line finder.

131

The interpretation tree then distinguishes aspect S2 from the rest by determining whether a neighboring region exists having the same moment size and direction around the target region. The interpretation tree tries to find such a region. In this case it succeeds, as shown in Figure 7 (c). From this, the interpretation tree determines that the target and the neighboring regions come from the same object and belong to the aspect S2.

The rest of the processing is to verify the determined aspect and to calculate accurate object attitude, again following the interpretation tree. Comparison of the EGIs from the model and the scene determines the viewer direction (Figure 7 (d)). Next, the viewer rotation around the viewer direction must be determined. From the relationship between the two regions, the viewer rotation can be determined up to two directions (180° apart) (Figure 7(f)), but more detailed analysis of the edge distribution is necessary to determine it uniquely. The interpretation tree examines the edge distributions in the two stripe regions which are predicted from the two possible rotations. This prediction can be obtained by applying the affine transform already established for this case. In this way, by following the interpretation tree as shown by the bold line (Figure 7(e)), the object has been recognized and its attitude has been calculated uniquely (Figure 7(g)). Figure 7 (h) presents the recognition result by projecting the object model with the detected attitude on top of the depth map.

For different aspects, other parts of the interpretation tree are similarly executed. When the interpretation tree has been executed on various regions in an image for another scene, the combined interpretation results look like Figure 8, in which 10 instances of objects have been located successfully.

3. Toward Systematic Methods of Compilation

The system presented in the previous section has compiled the object model into a recognition strategy in the form of an interpretation tree, and the resultant interpretation tree was successfully used to recognize the object instances in a cluttered bin-of-parts scene. In the off-line compilation stage, it automatically derived distinctive aspects from a geometrical object model, built feature descriptions of aspects by calculating expected feature values from the object model, and then, based on those descriptions, generated an interpretation tree for classifying the aspects and determining the attitude within each aspect. At on-line run time, the interpretation tree has controlled the localization process by using the predesignated most appropriate features
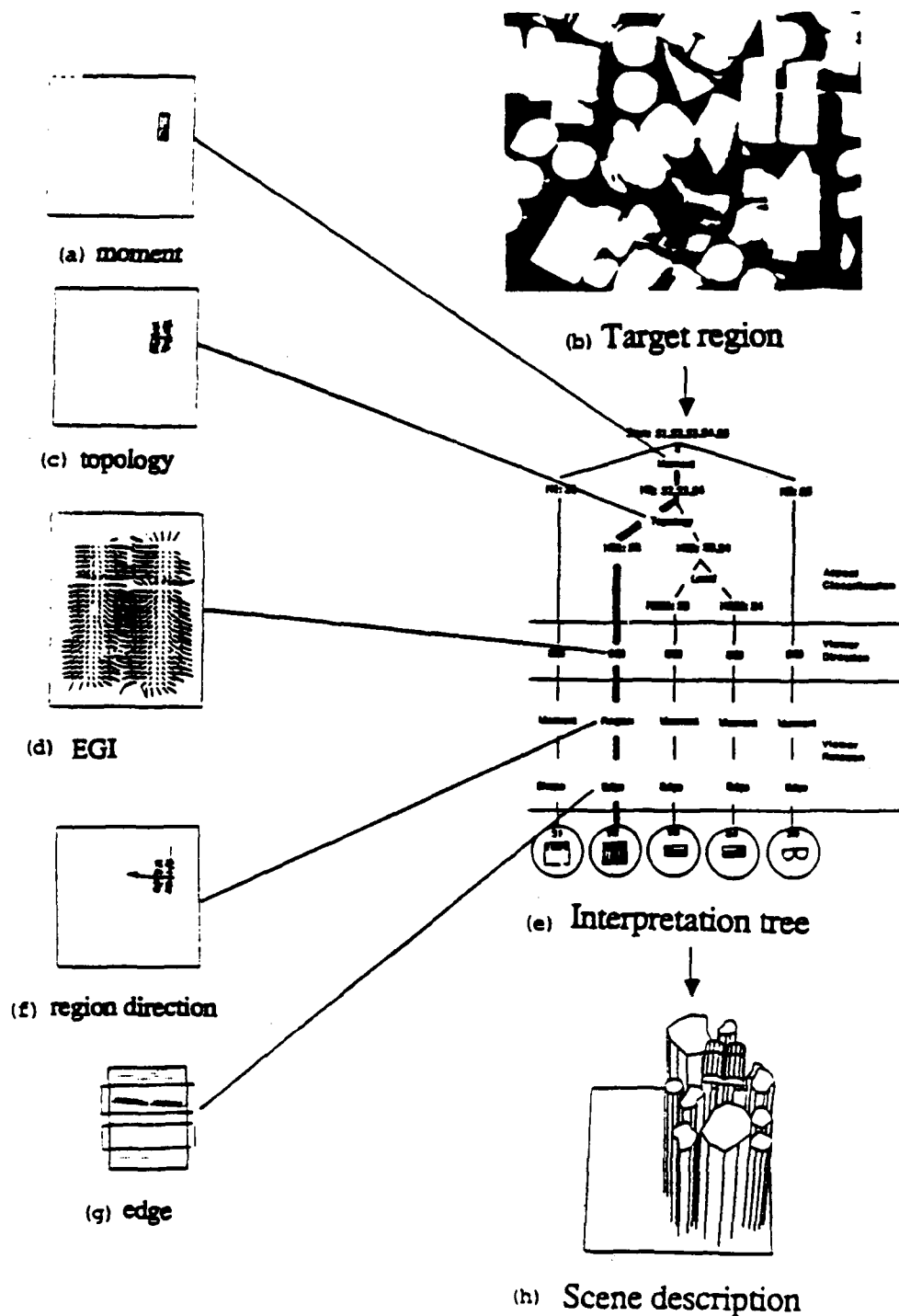
(a) moment

(b) Target region

(c) topology

(d) EGI

(e) Interpretation tree

(f) region direction

(g) edge

(h) Scene description

Figure 7: Execution of the interpretation tree: (a) Moment of the target region which is represented by a rectangle; (b) Target region. The white arrow indicates the the target region; (c) Neighboring region which belongs to the same object. From this evidence the interpretation tree determines that the target region and the neighboring region come from the same object and belong to the aspect S2; (d) EGI; (e) Interpretation tree. By following the interpretation tree as shown by the bold line, the object has been recognized and its attitude has been calculated uniquely; (f) Region direction. From the relationship between the two regions, the viewer rotation can be determined up to two directions (180° apart); (g) Edge distribution. The interpretation tree examines the edge distributions at locations and orientations predicted from the two possible rotations. This prediction can be obtained by applying the affine transform already established for this case to the edge representation in the aspect model; (h) Scene description.

133

(a)
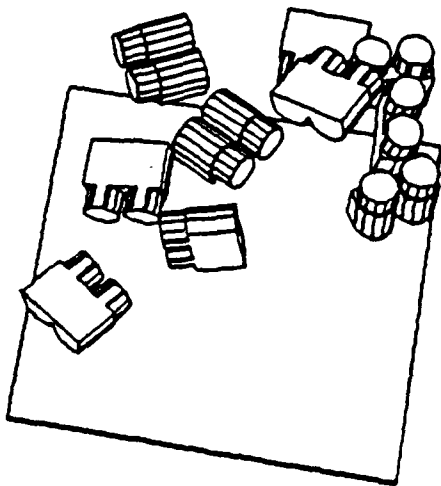
(b)
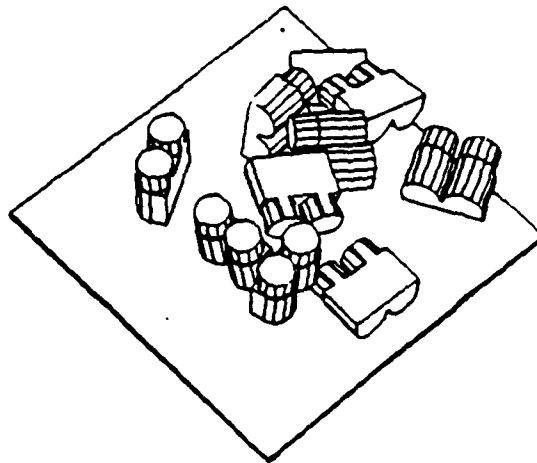
(c)

Figure 8: Another interpretation result: (a) Input scene (Top view); (b) Recognition result (Frontal view); (c) Recognition result (Side view).

at each stage. The recognized object position and attitude could be used for such tasks as bin-picking.

Though successful and promising, the system raises several important issues to be solved in order to develop a more systematic and general method of compiling recognition programs from models. We have found that one of the most crucial things is a more systematic way for modeling object appearances. So far, modeling has concentrated primarily on a geometric modeling of an object. Modeling ranges from generic models, such as generalized cylinders [6, 59], extended Gaussian images [31, 30], and superquadric models [55], to specific models such as aspect models [39, 57] region-relation models [4] and smooth local symmetry models [10]

However, the appearance of an object in an image, and the features of an object that can be reliably detected are determined not only by object properties, but also by sensor characteristics. As shown in Figure 9, the same object model in the same attitude can create different appearances and features when seen by different sensors. Edge-based binocular stereo reliably detects depth at edges perpendicular to the epipolar lines. Photometric stereo or a light-stripe range finder detects surface orientation and depth of surfaces which are illuminated and visible both by the light sources and by the camera.

Thus, in model based vision, it is insufficient to consider only an object model; it is essential to appropriately model sensors as well. Modeling sensors for model-based vision, however, has attracted little attention. In fact, the lack of explicit sensor models was the basic reason that the system in the previous section required human assistance. In order to make automatic and correct decisions, the system must correctly characterize object's appearances for the particular sensor in use, predict uncertainty ranges of feature values, and develop a framework to convert those predictions into decision rules. In the following sections, we will discuss some of the issues toward this goal, including representation of sensor-object relationships, characterization of detectability and reliability of sensors, prediction of uncertainty ranges of feature values, and generation of flexible execution programs.
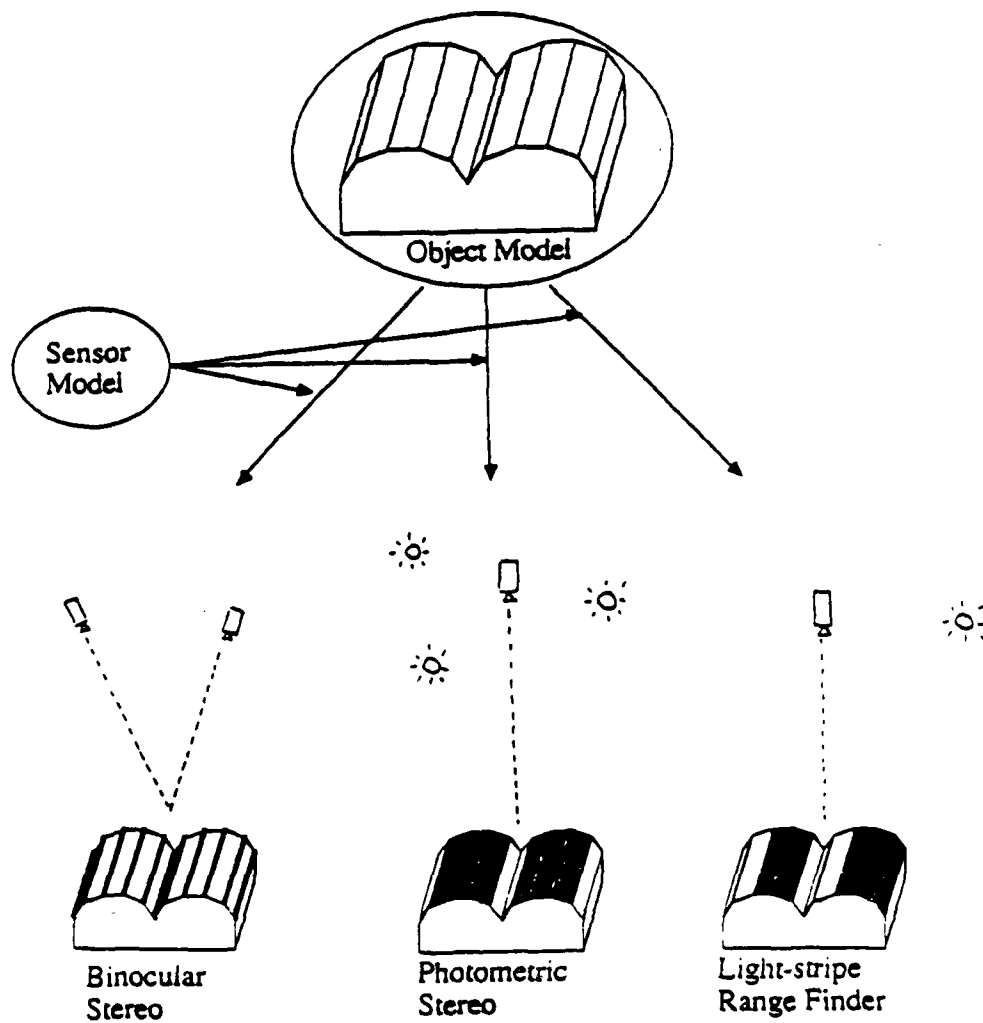
Figure 9: Object appearances by three different sensors. Edge-based binocular stereo reliable detects depth at edges perpendicular to the epipolar lines. Photometric stereo or a light-stripe range finder detects surface orientation and depth of surfaces which are illuminated and visible both by the light sources and by the camera. The same object model in the same attitude can create different appearances and features when seen by three different sensors.

4. Modeling Sensors

Different types of sensors are used in model-based vision. For our purpose, "sensors" are transducers which transform "object features" into "image features". For example, an edge detector detects edges of an object as lines in an image. Photometric stereo measures surface orientations of surface patches of an object. There are both passive and active sensors. Binocular stereo is passive, while a light-stripe range finder is an active sensor using actively controlled lighting. Table 1 gives a summary of various sensors in terms of what object features are detected in what forms.

Table 1: Summary of Sensors

Sensor	Vertex	Edge	Face	active/passive
Edge Detector [58, 45, 15]	-	line	-	passive
Shape-from-shading [29, 36]	-	-	region	passive
Synthetic Aperture Radar [19, 63, 48]	point	point/line	point	active
Time-of-Flight Range Finder [38, 27]	-	-	region	active
Light-stripe Range Finder [1, 54]	-	-	region	active
Binocular Stereo [46, 24, 2, 52]	-	line	-	passive
Trinocular Stereo [49]	-	line	-	passive
Photometric Stereo [64, 35]	-	-	region	active
Polarimetric light detector [42, 43]	-	-	point	active

In addition to qualitative descriptions of a sensor, a sensor model must model two characteristics quantitatively: *detectability* and *reliability*. Detectability specifies what kind of features can be detected in what conditions. Reliability specifies the expected error in the value of a feature. Since these two characteristics depend on how the sensor is located relative to an object feature, we will first define a feature configuration space to represent the geometrical relationship between the sensor and the feature. Then, we will investigate the way to specify detectability and reliability over the space.

4.1. Feature Configuration Space

Whether and how reliably a sensor detects an object feature depend on various factors: distance to a feature, attitude of a feature, reflectivity of a feature, transparency of air, ambient lighting, and so forth. In most model-based vision problems, the attitude of a feature, that is, angular freedom in the relationship between a feature and a sensor, affects sensor characteristics most. For that purpose, we attach a coordinate system to an object feature and consider the relationship between the sensor coordinate system and the feature coordinate system. For example, for a face feature, we define a coordinate system so that the z axis of the feature coordinate system agrees with the surface normal and x-y axes lies on the face, but defined arbitrarily otherwise. For other features, we define can feature coordinates appropriately.

For the sake of convenience let us fix the sensor coordinate system and discuss how to specify feature coordinates with respect to it. The angular from the sensor coordinate system to a feature coordinate system can be specified by three degrees of freedoms: two degrees of freedom in the direction of the z axis, and one degree of freedom in the rotation about the z axis. See Figure 10 (a).

We will define a sphere in which a feature coordinate system is specified as a point. Referring to Figure 10 (b), the direction from the sphere center to the point coincides with the z axis of the feature coordinate. The distance from the spherical surface to the point is determined by the angle of rotation (modulo 360°) around the i(z) axis from the coordinate on the spherical surface. A point on the spherical surface represents a feature coordinate obtained by rotating the sensor coordinate around the axis perpendicular to plane given by the sphere center, the spherical point, and the north pole. The north pole of the sphere is made to correspond to the case when the feature coordinate is aligned completely with the sensor coordinate.[6] We will refer to this sphere as the feature configuration space.[7]

---

[6] This representation will not create discontinuities around the north pole as opposed to the case in which Euler angles from the sensor coordinate frame to the feature coordinate frame are used to specify spherical points; this representation will instead create discontinuities at the center of the sphere and at the south pole. However, this is advantageous because we mostly use the area around the north pole to discuss detectability and reliability.

[7] Note that this sphere is different from the Gaussian sphere used in the previous section. Previously, the Gaussian sphere represented the sensor coordinates (the viewer directions) with respect to the object coordinates and detectability of each feature was examined by an *adhoc* method for each viewer direction. In contrast, here we are developing a tool to examine the detectability of a feature using the sphere to represent the feature coordinates with respect to the sensor coordinates. This tool will be applied to features of an object which is rotated with respect to the sensor coordinates.
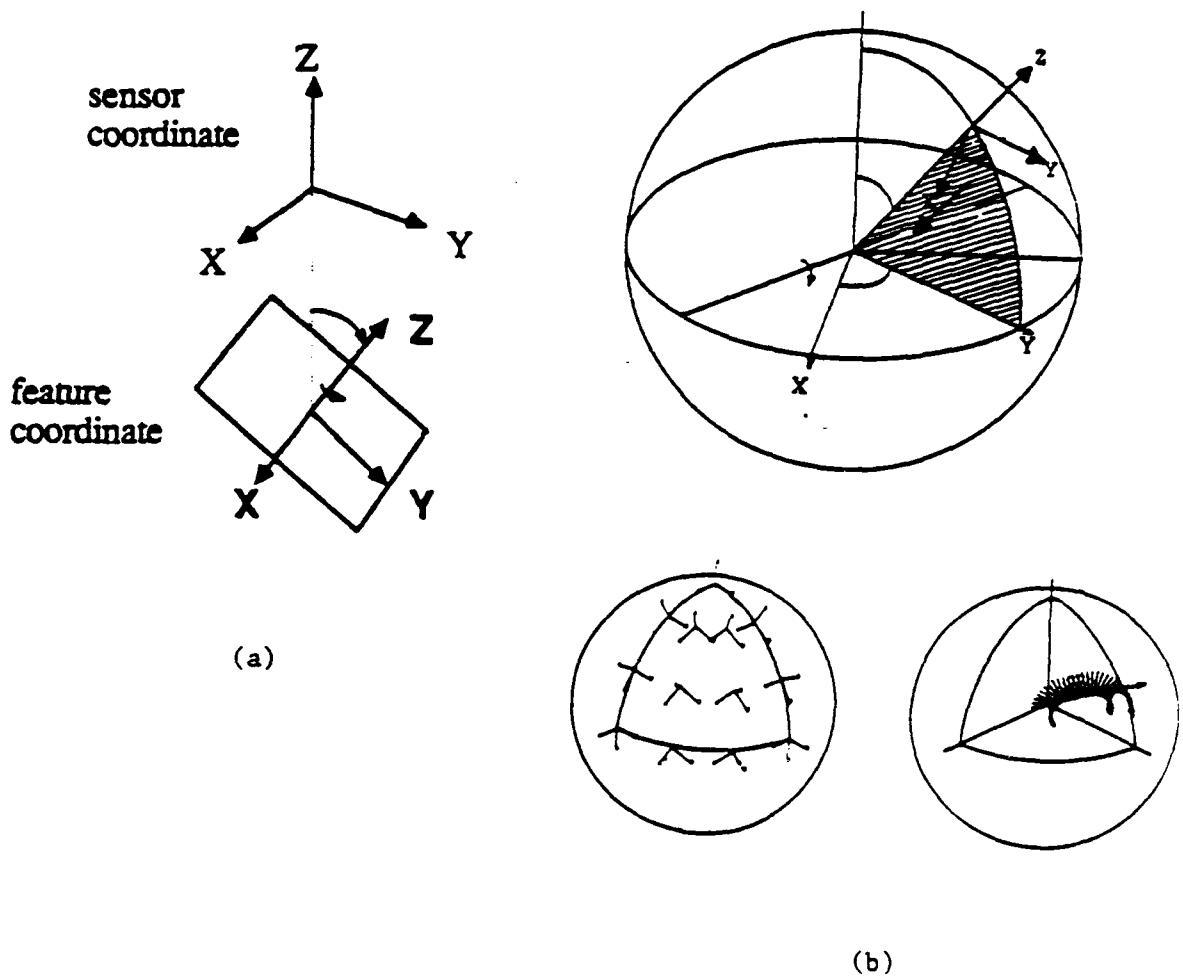
(a)

(b)

Figure 10: Feature configuration space: (a) Relationship between sensor coordinate and feature coordinate. The feature coordinates can be specified by three degrees of freedoms: two degrees of freedom in the direction of the z axis of a feature, and one degree of freedom in the rotation about the z axis of a feature. (b) Feature configuration space. One feature coordinate can be represented as a point in the sphere. The direction from the sphere center to the point coincides with the z axis of the feature coordinate. The distance from the spherical surface to the point id determined by the angle of rotation (modulo 360°) around the feature z axis from the coordinate on the spherical surface. A point on the spherical surface represents a feature coordinate obtained by rotating the sensor coordinate around the axis perpendicular to the plane given by the sphere center, the spherical point, and the north pole. The drawing at the bottom left depicts the coordinates corresponding to the points on the spherical surface, while the one at the bottom right depicts the coordinates corresponding to the points on one axis.

139

4.2. Constraints on Feature Detectability

Using the feature configuration space, we will represent in a general way the constraints on the attitude of a feature for it to be detected by a sensor. A sensor has two types of components in general: illuminators and detectors. In order for a feature to be detected by a sensor, it must satisfy certain conditions on being illuminated by its illuminators and being visible from its detectors.

Once we define a local coordinate system on an object feature, we can compute configurations of a feature in which it is illuminated by each illuminator, and configurations in which it is visible by each detector. In this analysis, it should be noted that illuminators and detectors can be treated interchangeably. In [37] this concept was defined as generalized sources (G-sources). The illumination direction of a illuminator and the line of sight of a detector correspond to the G-source illumination directions, and both can be represented in the feature configuration space as a radial line from the sphere center. Also, illuminated configurations by an illuminator and visible configurations from a detector correspond to the G-source illuminated configuration, and both can be specified as a volume in the configuration space. Finally, we can obtain the constraints in which the feature is detectable by the sensor with AND and OR operations on illumination (line-of-sight) directions and illuminated (visible) configurations of all components of sensors.

Figure 11 shows an example analysis of a face feature for a light-stripe range finder. A light-stripe range finder has two G-sources (a TV camera and a light source): the direction denoted by $V1$ indicates the line of sight of the TV camera; $V2$ indicates the illumination direction of the light source. The illuminated configurations of a face are determined by the z axis (ie, its surface normal), and are not dependent on its rotation. Therefore, illuminated configurations of a feature form a spherical cone whose axis is $V2$ and whose apex angle is $d2$. Similarly, the configurations of a feature visible from the TV camera form a spherical cone whose center direction is $V1$ and whose apex angle is $d1$. Since a light-stripe range finder detects the faces which are illuminated from the source and visible from the TV camera, the detectable configurations are the intersection of the two cones. Similarly we can analyze the detectable configurations of various features for various sensors in Table 1. The results of the analysis are summarized in Figure 12: for more details, see [37].
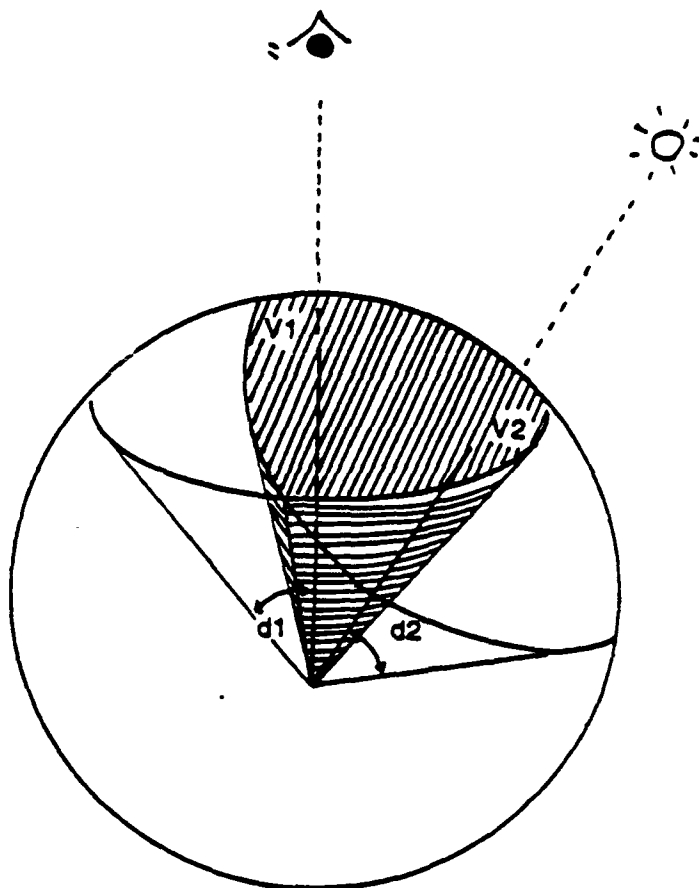
Figure 11: Detectability configurations of a face for a light-stripe range finder.
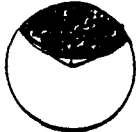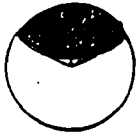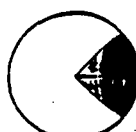
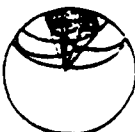| Sensor	Constraints in the formal definition	Constraints in the sensor space	Sensor	Constraints in the formal definition	Constraints in the sensor space
Edge Detector	(AND (NS edge V d) (NS edge V d)) = (NS edge V d)		Binocular Stereo	(AND (NS edge V1 d1) (OS edge V2 d2 VE de) (OS edge V3 d3 VE de))	
Shape-from-shading	(AND (NS face V d) (NS face V d)) = (NS face V d)		Trinocular Stereo	(AND (NS edge V1 d1) (OS edge V2 d2 VE de) (OS edge V3 d2 VE de) (OS edge V4 d2 VE de))	
SAR	(OR (NS face V d) (NS edge V d) (NS vertex V dj) (needs postprocess)		Photometric Stereo	(AND (NS face V d1) (NS face V1 d2) (NS face V2 d2) (NS face V3 d2))	
Time-of-Flight Range Finder	(AND (NS face V d) (NS face V d)) = (NS face V d)		Polarimetric Light Detector	(OR (AND (NS face V d) (NS face V1 d)) (AND (NS face V d) (NS face V2 d)) ...) where V . V = cos 2d	
Light-strip Range Finder	(AND (NS face V1 d) (NS face V2 d))				

Figure 12: Summary of detectability configurations for various sensors. The feature coordinate of a face is defined so that z axis agrees with the surface normal and the x-y axes lie on the face. The feature coordinate of an edge is defined so that z axis agrees with the direction of $\frac{N_1+N_2}{2}$, where N_1, N_2 are normal vectors of two incident faces to the edge. x axis agrees with the edge direction.

5. Modeling Appearances

Aspects have been defined as topologically equivalent classes with respect to the object features "visible" to the sensors. Classifying object appearances into aspects systematically raises several issues. First, since aspect is defined relative to sensors, the detectability of features by the particular sensors to be used must be incorporated. In the system of section 2, however, we used the constraints of the surface visibility by the photometric stereo in an *adhoc* manner. Now that we have developed a way to represent the detectable configurations of features, we can use it in generating appearances. Second, we will discuss how to represent object appearances and aspects in a systematic way. In the previous system, output from the geometric modeler is handled by a human-assisted process to analyze them and to generate a recognition strategy from them. This interactive process can handle any *adhoc* representations. However, in the present system, a complete automatic process should handle the output and generate a recognition program. This requires a systematic representation of object appearances as well as aspects. Third, transition from one aspect to another may not be a discrete process because the detectability of features tends to degrade near the boundary of detectable configurations. Finally, it is useful to obtain an estimate on the number of aspects in order to make sure that the recognition methods based on aspects are applicable to an object with a reasonable complexity. This section will discuss these four issues.

5.1. Appearance Generation from Constraints on Feature Detectability

To predict object appearances, we apply the constraints on feature detectability to each feature of the object. Each feature is detectable by the sensor if it satisfies the following two conditions:

1. None of the illumination (line-of-sight) directions are occluded by any other parts of the object;

2. The detectable configurations contain the configuration of the feature.

To check these conditions we use the constraints together with a geometric modeler. We rotate the object into a certain attitude to be examined, and then see whether its features satisfy the previous constraints.

Figure 13 illustrates this process of predicting object appearances for a light-stripe range finder. Suppose an object is placed like Figure 13 (a). Figure 13 (b) shows the detectability constraints on a face for a light-stripe range finder. We will put this configuration space on each candidate face to examine whether the face is detectable. See Figure 13(c). This amounts to

143

checking the following conditions:

1. The light source direction is not occluded by other faces.

2. The line of sight of the TV camera is not occluded by other faces.

3. The local coordinate of an face, defined by the surface orientation (z axis) and the tangential plan (x-y axis), is contained in the detectable configurations.

Figure 13(d) shows the result of this operation. The shaded areas indicate those which satisfy the conditions and thus are detectable by the light-stripe range finder.

5.2. Describing Aspects

Appropriate descriptions of aspects must be defined so that they can be used in automatic generation of interpretation trees. The description of an aspect should include constituent appearances, a set of features extractable for the aspect, and the expected feature values. This description should have flexible and convenient forms for applying generation rules to them and for use in execution. We will represent aspects on frames by using a frame representation language, Framekit+, because it has a flexible structure and powerful demon facilities. Since an aspect is an abstract concept which represents a group of possible appearances, we will first consider how to represent each appearance in the frame. Then, we will represent aspects based on the representation of appearances.

A geometric modeler generates a possible appearance of an object under a given attitude. We will convert output data from the geometric modeler into representations in Framekit+. One appearance, for example 10 in Figure 14(a), is represented by one frame, which points to several appearance component frames representing visible 2D faces, *IMAGE-COMP01*, and *IMAGE-COMP02*[8]. Each frame corresponding to one visible 2D face maintains various geometric properties of the face in slots. For example, face area and face moment are maintained in slots *AREA* and *MOMENT*. The values of these features are obtained by using output data from a geometric modeler. Each frame representing a 2D visible face has a backpointer to the 3D face from which the 2D face is projected. For example, the

---

[8] In this example, one 2D face corresponds to one image component. If several 2D faces have C^1 continuity across the edges, these faces are grouped and stored as one single image component. In this case, face area and face moment are calculated over the group of faces.
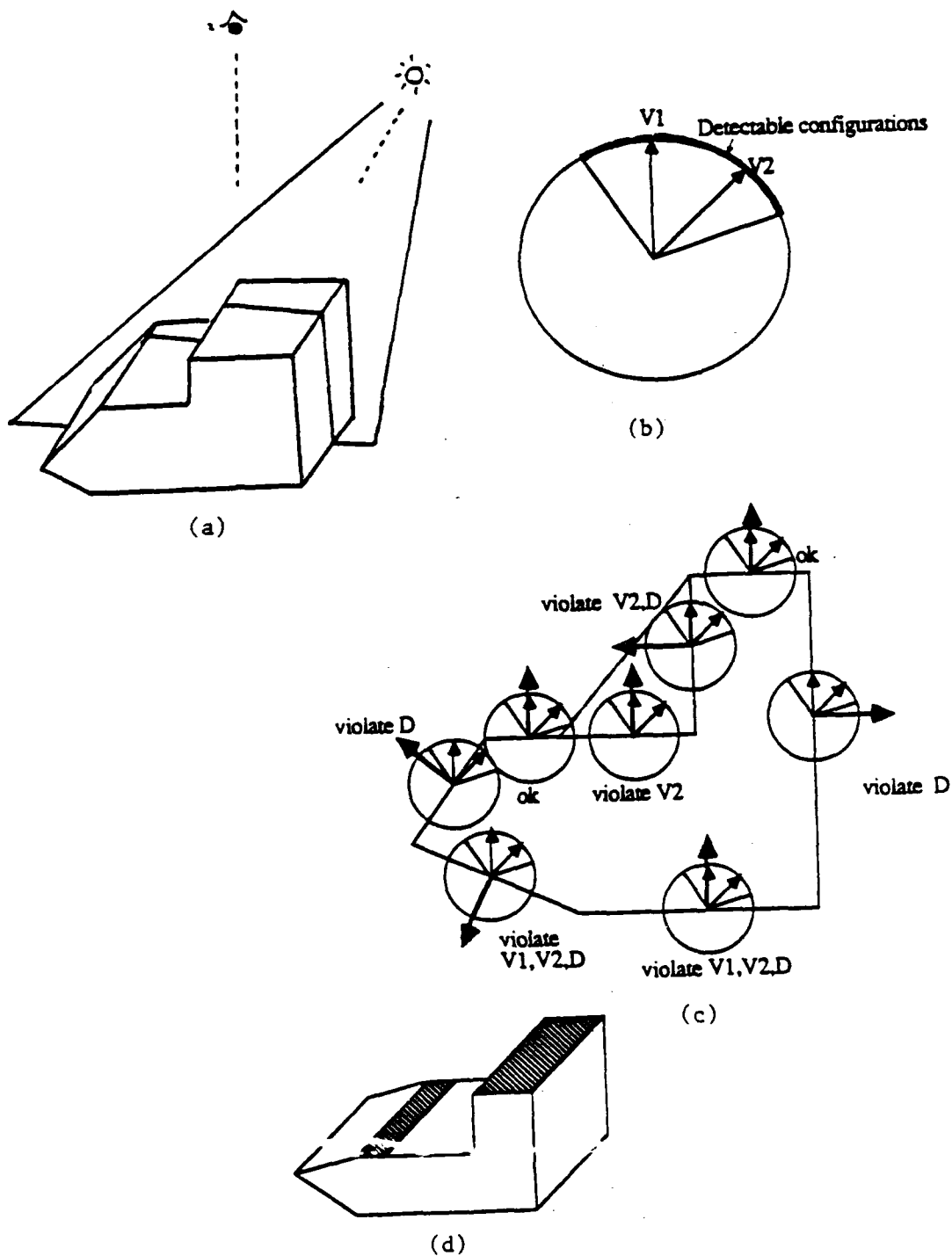
Figure 13: How to use detectability configurations: (a) Light-stripe range finder; (b) Detectability constraints on a face for a light-stripe range finder. The constraints consist of detectable configurations and two G-source illumination directions, $V1, V2$; (c) Appling detectability configuration; (d) Detectable faces. The shaded area indicate those which satisfy the conditions and thus are detectable by the light-stripe range finder.
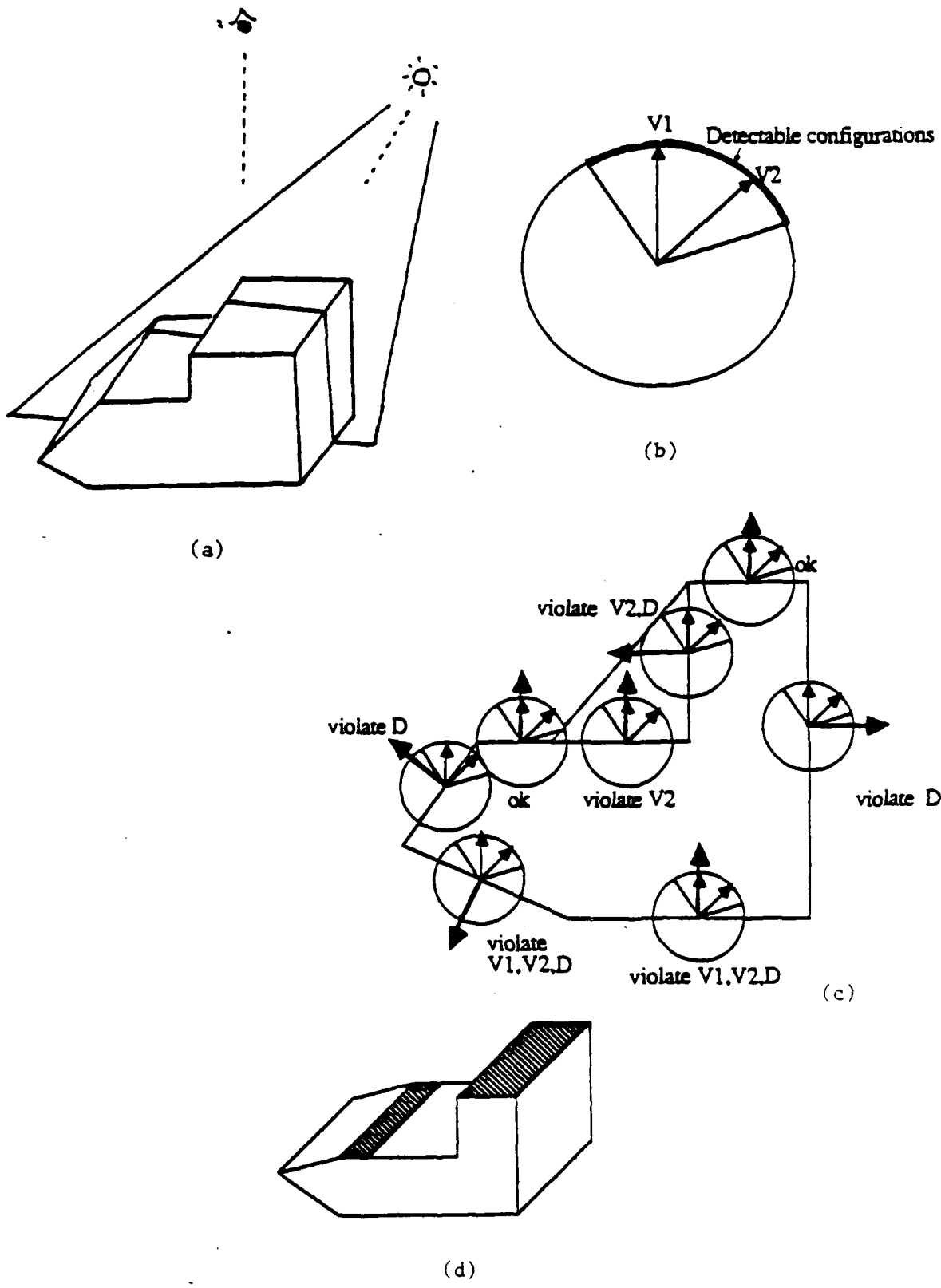
Figure 13: How to use detectability configurations: (a) Light-stripe range finder; (b) Detectability constraints on a face for a light-stripe range finder. The constraints consist of detectable configurations and two G-source illumination directions. *V1,V2*; (c) Appling detectability configuration; (d) Detectable faces. The shaded area indicate those which satisfy the conditions and thus are detectable by the light-stripe range finder.

IS-A-FACE-OF slot of *IMAGE-COMP01* frame has a value *FACE6*.[9] An image structure consists of an image frame and image component frames.

Once image structures are represented, we can generate aspect structures in frames. Since an aspect is an abstract concept for a group of images (appearances), an aspect structure is similar to its constituent image structures. In order to construct aspect structures, shape labels of all image frames are examined one by one, where a shape label is the combination of visible 3D faces as explained in section 2.1. The visible 3D faces among a 2D appearance can be retrieved by backpointers of 2D faces to 3D faces such as *FACE6* in *IS-A-FACE-OF* slot of *IMAGE-COMP01* frame, where *FACE6* is the frame name of a 3D face of the object.

If an image structure cannot find any aspect structure with the same shape label among the already established ones, a new aspect frame is created together with aspect component frames which correspond to image component frames: therefore, the aspect structure has the same structure as the image structure. Also, frames to represent the relationships between pairs of aspect components are created. If an image structure can find an aspect structure with the same shape label, the image frame is registered to the aspect frame as an instance and its frames of 2D faces are registered to corresponding aspect component frames.

An example of an aspect structure is shown in Figure 14(b). Aspect frame *ASPECT1* points to several aspect component frames, *ASPECT-COMP10*, *ASPECT-COMP11* with the *IS-AN-ASPECT-COMP-OF+INV* slot. It also points to its instance images *I0*, *I1* with *IS-AN-IMAGE-OF-ASPECT-OF+INV* slot, while its aspect component frame, *ASPECT-COMP10* points to its instance 2D faces *IMAGE-COMPU1*, *IMAGE-COMP12*. Frame *ASPECT-COMP-RELATION-11-10* is a relation frame which represents the relationship between *ASPECT-COMP10* and *ASPECT-COMP11*.

---

[9]Each frame also contains array addresses of various geometric items such as 2D FACE, 2D EDGE and 2D VERTEX in the data base of the geometric modeler; for example, *9361* in *REGIONS* slot of *IMAGE-COMP01* frame. These allow us to access the original geometric data, if necessary.
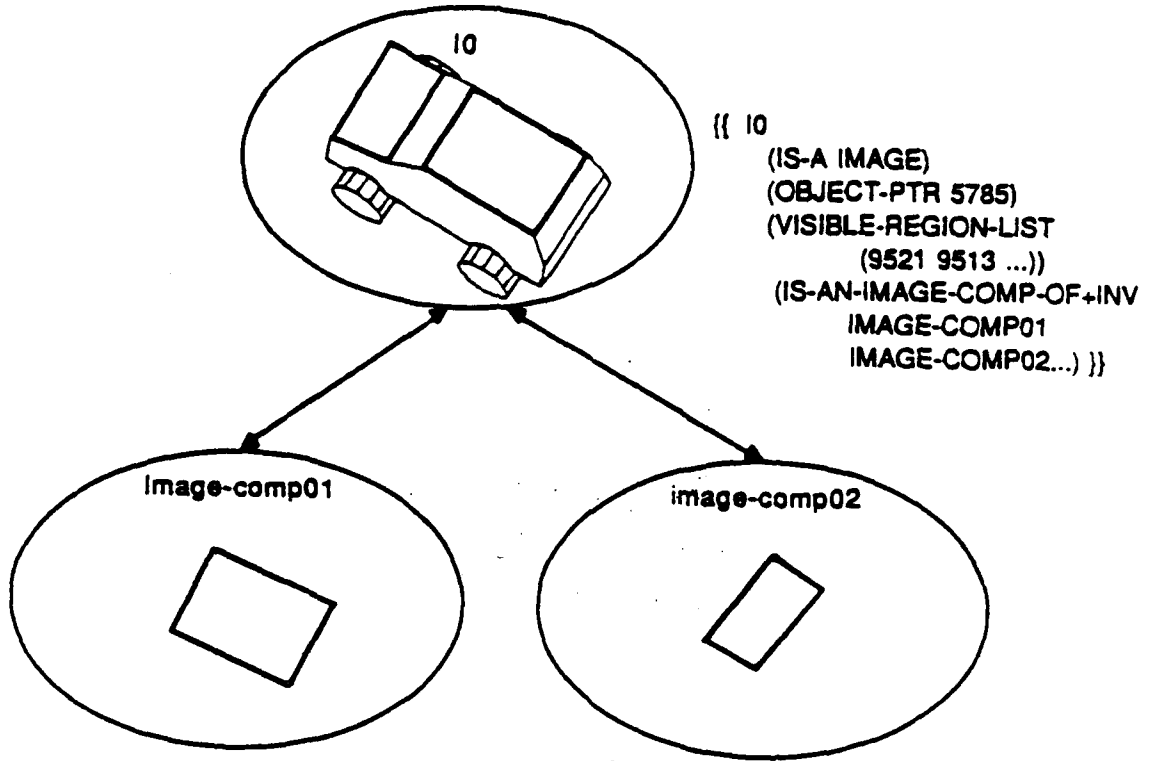
```
{{ 10
   (IS-A IMAGE)
   (OBJECT-PTR 5785)
   (VISIBLE-REGION-LIST
        (9521 9513 ...))
   (IS-AN-IMAGE-COMP-OF+INV
        IMAGE-COMP01
        IMAGE-COMP02...) }}
```

```
{{ IMAGE-COMP01
   (IS-A  IMAGE-COMP)
   (IS-AN-IMAGE-COMP-OF 10)
   (IS-A-FACE-OF FACE6)
   (REGIONS (9361))
   (AREA 13.88)
   (MASS-CENTER (1.21 0.24))
   (MOMENT (22.50  11.47 -0.53))
   (NORMAL (0.0001 0.355 0.935)) }}
```
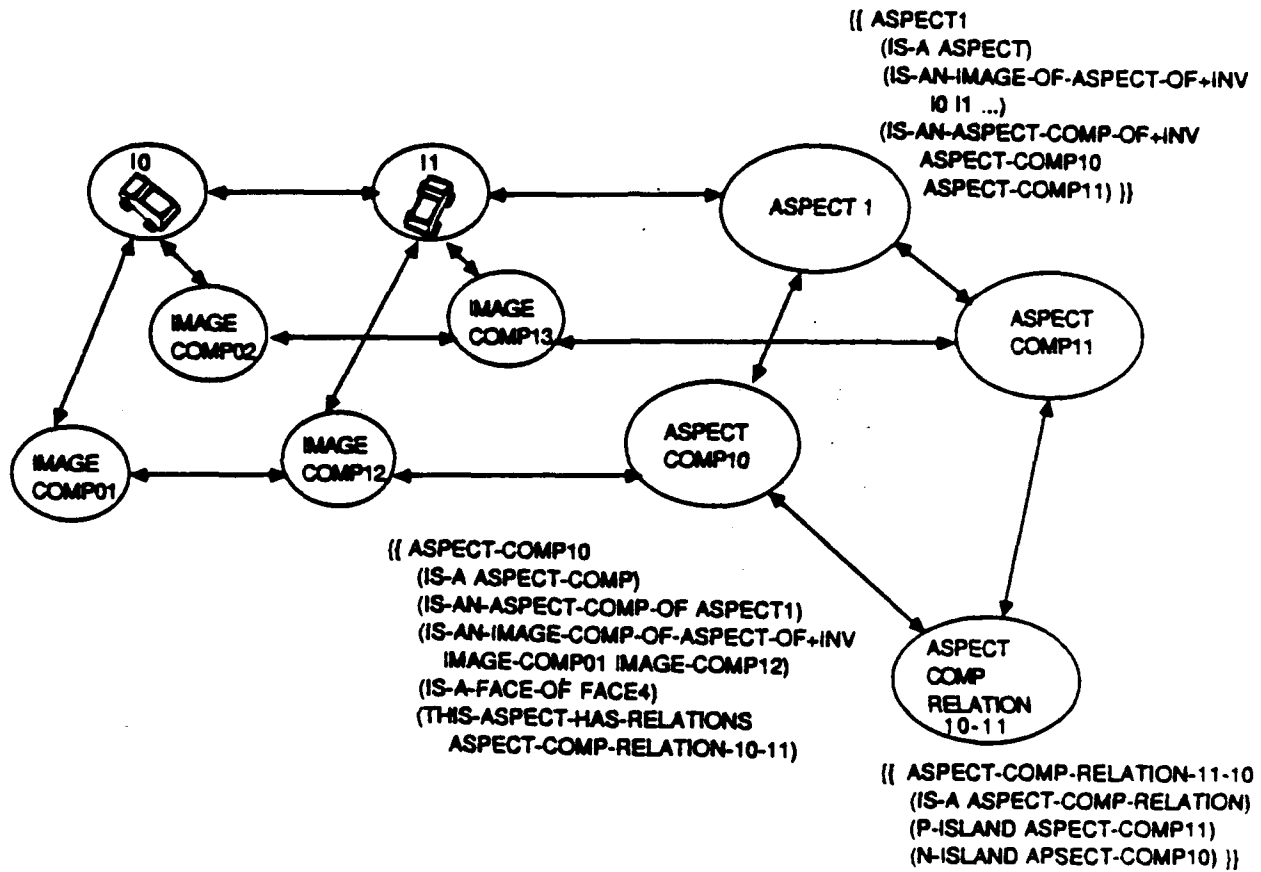
```
{{ IMAGE-COMP02
   (IS-A  IMAGE-COMP)
   (IS-AN-IMAGE-COMP-OF 10)
   (IS-A-FACE-OF FACE4)
   (REGIONS (9481))
   (AREA  7.47)
   (MASS-CENTER (-2.50 2.38)
   (MOMENT (8.56 2.60 0.80))
   (NORMAL (-0.17 0.46 0.87)) }}
```

(a)

143

((ASPECT1
  (IS-A ASPECT)
  (IS-AN-IMAGE-OF-ASPECT-OF+INV
    I0 I1 ...)
  (IS-AN-ASPECT-COMP-OF+INV
    ASPECT-COMP10
    ASPECT-COMP11)))

((ASPECT-COMP10
  (IS-A ASPECT-COMP)
  (IS-AN-ASPECT-COMP-OF ASPECT1)
  (IS-AN-IMAGE-COMP-OF-ASPECT-OF+INV
    IMAGE-COMP01 IMAGE-COMP12)
  (IS-A-FACE-OF FACE4)
  (THIS-ASPECT-HAS-RELATIONS
    ASPECT-COMP-RELATION-10-11)

((ASPECT-COMP-RELATION-11-10
  (IS-A ASPECT-COMP-RELATION)
  (P-ISLAND ASPECT-COMP11)
  (N-ISLAND APSECT-COMP10)))

(b)

Figure 14: Frame representation of aspects: (a) Image structure. Each image structure consists of a frame corresponding to an image and several frames corresponding to 2D visible faces in the image; (b) Aspect structure. Each aspect structure consists of an aspect frame, aspect component frames, and aspect component relation frames.

5.3. Probability Distribution of Detectability and Transition of Aspects

So far, we have treated sensor detectability as a discrete process: detectable and undetectable. Thus, aspect changes occur abruptly. Actually, however, sensor detectability is a continuum, so aspect changes occur continuously. The detectable configurations in the space give the limit of detectability. Near the boundary, however, even when an object feature exists within the detectable configurations, it may be undetectable due to noise. We will investigate how the detectability varies probabilistically over the detectable configurations.

For an example, let us consider a hypothetical light-strip range finder. A light-stripe range finder projects a plane of light onto the scene and determines the position of a surface patch from the slit image. The detectability depends on whether the brightness of the slit image is bright enough to be detected, say brighter than a threshold I_0. Assuming a Lambertian surface, the brightness of the slit image is given by $I_s N \cdot S$ where N is the surface normal, S is the light source direction, and I_s is the light source brightness. If we assume an additive zero-mean Gaussian noise of brightness with power σ^2, the resultant brightness distribution of a slit will be

$$p(I) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(I - I_s N \cdot S)^2}{2\sigma^2}}.$$

Thus, the probability distribution of feature detectability of our hypothetical range finder can be described as

$$P_d = Prob(I \geq I_0) = \int_{I_0 - I_s N \cdot S}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{I^2}{2\sigma^2}} dI = \Phi(\frac{I_0 - I_s N \cdot S}{\sigma})$$

As shown in Figure 15, this probability decreases as the incident angle of the light stripe increases, and near the boundary of the illuminated configuration of the light source, the probability approaches 0.

This continuous change of detectability causes the continuous aspect transition and the aspect boundaries become blurred. In order to characterize an aspect boundary, we can define the distance between two aspects across the boundary by the Hamming distance between their corresponding shape labels $\{x_1, x_2, ..., x_i, ..., x_n\}$, where $x_i = 1$ if face i is visible and $x_i = 0$ otherwise. Thus, the distance of two aspects is the number of faces which switch between visible and nonvisible states across the aspect boundary.

Consider an aspect boundary between aspect A and B whose Hamming distance is one, that is, aspect A and B differ in visibility of only one face F_i. Suppose the detectability of face i is $P_d(F_i)$. Then, near the aspect boundary, the aspect A may be observed incorrectly as aspect B with probability $1-P_d(F_i)$. A similar false observation will also occur for aspect B.

If the distance of aspects A and B is more than one across a boundary, then erroneous intermediate aspects, which are neither A or B, can occur near the boundary. This can be easily seen by considering an example where aspect A has $\{x_i,x_j\}=\{10\}$ and aspect B has $\{x_i,x_j\}=\{01\}$ as shape labels, respectively. Then, we will observe object appearances belonging to four aspects near the boundary: aspects $\{11\}$ and $\{00\}$ in addition to aspects A and B. For example, the probability of observing aspect 11, instead of aspect A, is $P_d(F_i)P_d(F_j)$. This consideration must be taken into account when grouping and classifying aspects by an interpretation tree.

5.4. Estimating the Number of Aspects

An interesting and important question related to using aspects for object recognition is how many aspects an object will have. If this number is extremely large, it is impractical to classify an unknown scene into an aspect and then to determine the attitude within it.

One might think that the number of distinct aspects grows exponentially as the number of faces n in the object increases. However, the number of aspects grows much slower by a polynomial in n. To see this, let us consider the number of aspects $f_p(n)$ of a 2D convex polygon with n edges seen in perspective. The sensor can be placed at any point on the 2D plane. Each edge, when extended, divides the 2D plane into two half plane: when the sensor is located in the half plane corresponding to the front side of the edge, then the edge is visible; otherwise it is invisible. Therefore, the problem of obtaining the number of distinct aspects $f_p(n)$ is equivalent to obtaining the number of regions into which n lines divide a 2D plane. In fact, the visible/nonvisible combinations attached to each region make up the shape label.

We can derive the formula of $f_p(n)$ by an inductive method. Suppose we add the n-th line after $n-1$ lines have already been drawn. This new line intersects the existing $n-1$ lines at $n-1$ points (we are assuming the maximal case), which divide the new line into n segments. Each segment on the new line divides one old region into two regions. Thus, this operation adds n new regions. That is,

$$f_p(n) = f_p(n-1)+n.$$

By solving this, we obtain

$$f_p(n) = \frac{n^2+n}{2}+1.$$

as the upper bound on the number of aspects of a 2D convex polygon under perspective projection.

We can obtain the number of aspects $F_p(n)$ of a convex 3D polyhedron with n faces in a very similar way. In this case, each face, when extended, divides a 3D space into two 3D half spaces. We have to count the number of volumes that result when n planes divide a 3D space. We can again use an inductive method. Assume that we have divided the 3D space by $n-1$ planes. As shown in Figure 16, if we add the n-th plane, it intersects with the existing old $n-1$ planes, and generates $n-1$ intersection lines on it. Thus, on this n-th plane there are $f_p(n-1)$ polygons, each of which divides an old volume into two. Therefore, addition of the n-th plane adds $f_p(n-1)$ volumes:

$$F_p(n) = F_p(n-1)+f_p(n-1).$$

Thus,

$$F_p(n) = n^3/6+5n/6+1$$

is the upper bound on the number of aspects of a 3D convex polyhedron with n faces under perspective projection.

If we can assume orthographic projection, as we have done in our previous system, the number of aspects further reduces. Orthographic projection limits the possible sensor positions on the infinite sphere, and one occluding plane draws a great circle on the sphere to divide it into two hemispheres. We should count the number of regions on the sphere divided by n great circles. Since the n-th great circle intersect with the previous $n-1$ great circles at $2(n-1)$ points and adds $2(n-1)$ new regions, we obtain the following recursive equation:

$$F_o(n) = F_o(n-1)+2(n-1).$$

Thus,

$$F_o(n) = n^2-n+2.$$

We notice that the upper bounds of the number of aspects grows as a quadratic function of the number of faces n. Moreover, for practical recognition purposes, n should be taken as the number of significantly large faces rather than including all the tiny faces.

Non-convex polyhedra have more aspects, because aspects are determined not only by occluding planes due to faces but also occlusions due to edges. Suppose a 3D non-convex
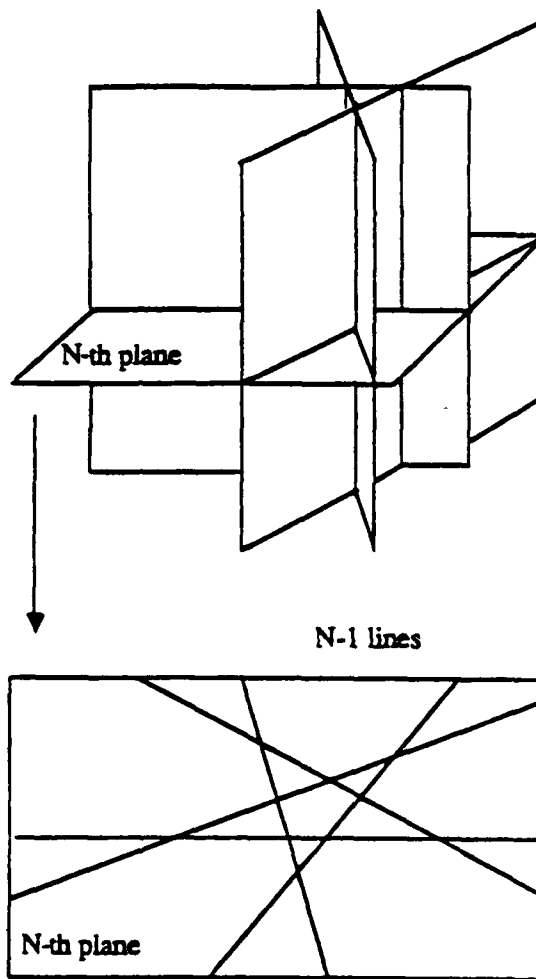
N-1 lines

N-th plane

Figure 16: Intersection of N planes

polyhedron has n faces, o edges, and p vertices. In the worst case, we have to consider occlusion planes defined by all the pairs of edge and vertex: $o \times p$. Thus, $F_o(n + o \times p)$ provides the upper bounds. However, in reality, the number of aspects must be much smaller, because a large fraction of pairs of vertex and edge either need not be considered or do not generate significant aspects to be taken into account for recognition.

6. Predicting Uncertainty in Feature Values

In classifying an unknown scene into an aspect and determining its exact attitude, we need to select features with high reliability and discriminant power. The reliability and discriminant power of a feature depend not only on the nominal value that the aspect is expected to have, but also its expected variances over the aspect. For example, imagine a geometric feature whose nominal values for two aspects are calculated as 100 and 90 by a geometric modeler. If a sensor has an uncertainty of plus/minus 1 for the feature, the feature is a reliable discriminator to separate the two aspects. On the other hand, if the uncertainty of the sensor is plus/minus 20, the feature is not usable. Therefore, prediction of the uncertainty that a feature will take over an aspect is very important for strategy generation.

This section will discuss a method to predict uncertainties of feature values. We must consider two levels of feature uncertainty. The first is the uncertainty in sensory measurements and this is obtained by analyzing the measurement mechanism of a sensor. In many cases, however, a geometric feature is derived from a set of sensory measurements and is used as a discriminator. We must also analyze the propagation of uncertainty from sensory measurements to a derived geometric feature in order to determine its uncertainty.

6.1. Uncertainty in Sensory Measurements

As an example of predicting the uncertainty of sensory measurements, we will again consider a depth measurement by a hypothetical light-stripe range finder. Let us assume that the main source of the depth uncertainty measurement by this sensor comes from the ambiguity of the slit position on a surface due to the width of the light beam and angular errors in setting the light directions. The error model can be obtained analytically.

As shown in Figure 17 (a), let us denote the angular ambiguity of the light stripe by $\delta\theta$. The light is intercepted by an object surface, creating a slit pattern on it. The angular ambiguity $\delta\theta$ of

the light direction results in ambiguity δy in the position on the surface:

$$\delta y = \frac{r\delta\theta}{\cos\alpha},$$

where r is the distance of the surface from the light source, and α is the angle between the light direction S and the surface normal N. This positional ambiguity on the surface is observed as the slit position ambiguity (or "slit width") δi in the camera image. If β is the angle between the surface normal N and the viewer direction V, then

$$\delta i = (\cos\beta)\delta y,$$

Finally, this ambiguity is transferred into the uncertainty in the depth measurement by triangulation. For simplicity, if we assume orthographic projection for the camera, the ambiguity in the image δi creates uncertainty in distance δz,

$$\delta z = \frac{\delta i}{\tan\gamma},$$

where γ is the angle between V and S.

In total, by representing the angles α, β, and γ in terms of V, N, and S, we obtain

$$\delta z = \frac{\cos\beta}{\cos\alpha\tan\gamma} r\delta\theta = \frac{(N\cdot V)(S\cdot V)}{(N\cdot S)\sqrt{1-S\cdot V}} r\delta\theta.$$

Since r is roughly constant, the uncertainty distribution of this light-stripe range finder over the detectable area is governed by the factor $\frac{(N\cdot V)(S\cdot V)}{(N\cdot S)\sqrt{1-S\cdot V}}$. Figure 17 (b) plots this function.

6.2. Uncertainty in Geometric Features

Usually sensory measurements, such as depth detected by a sensor are further converted into object features such as area and moment of a face. This process involves grouping pixels into regions, extracting some feature values and transforming them into another. Modeling the uncertainty generation and propagation in this process is difficult in general, but as an example of predicting uncertainty in a geometric feature, let us consider an area feature of a face detected as a region by our hypothetical light-stripe range finder. Figure 18 shows the conversion process from depth values to the area of a face. The process includes three parts: obtain the area of the corresponding region in the image, compute the surface orientation of the region, and finally convert the image area into the surface area by the affine transform determined by the surface orientation. We will analyze how uncertainty is introduced and propagated in these three parts.

Suppose that a surface under consideration has the real area A and the surface orientation β

155
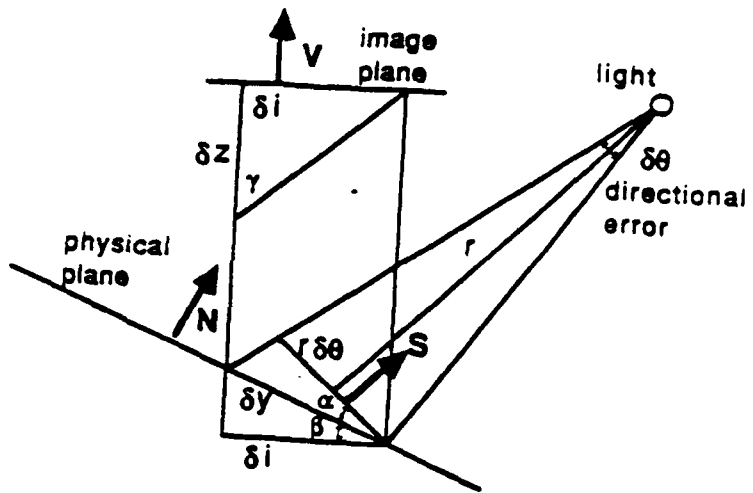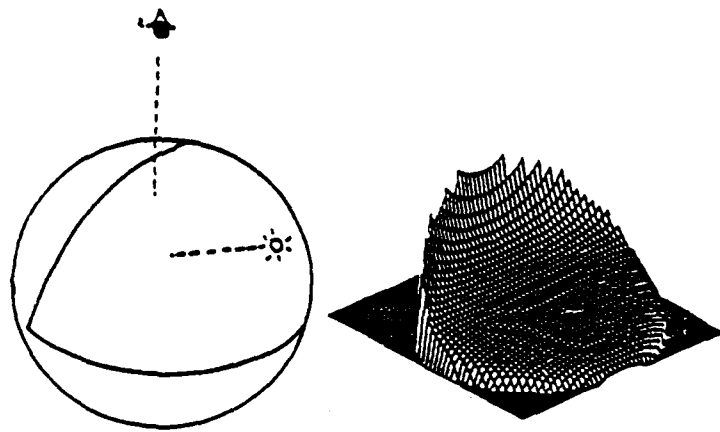
Figure 17: Predicting uncertainty in a sensory measurement by a light-stripe range finder: (a) Detection mechanism; (b) Predicted error range of a depth measurement.
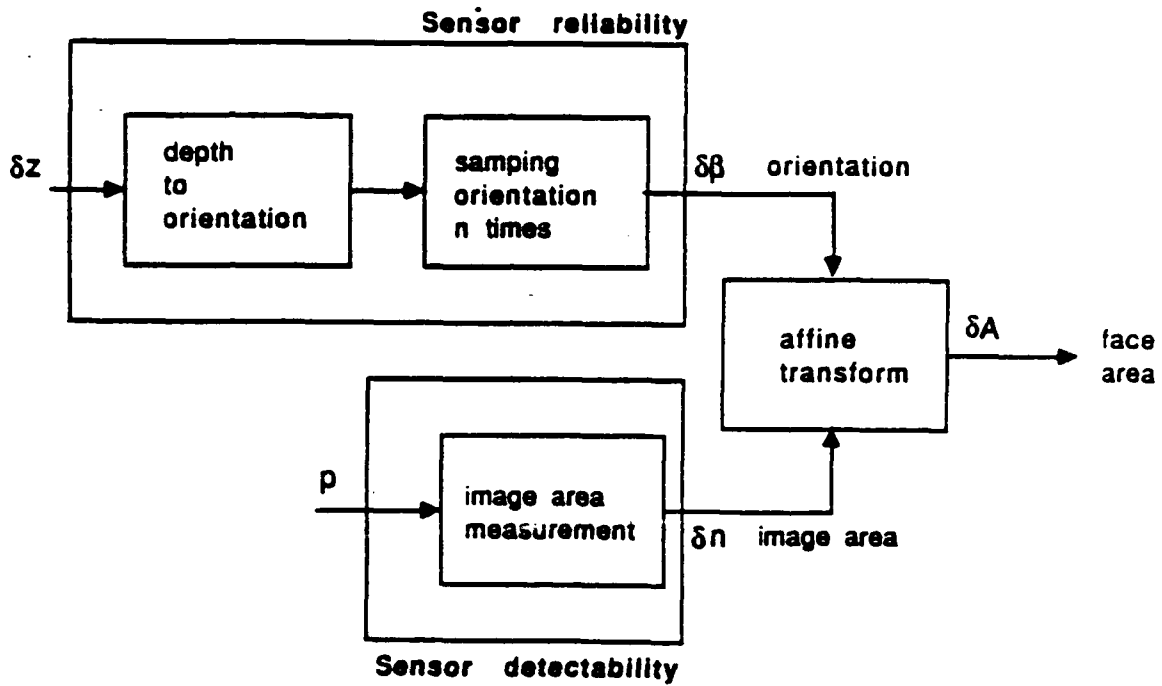
Figure 18: Conversion process from depth value values to the area of a face.

(angle between the surface normal and the viewing direction). It should create a region of size n pixels where

$$n = A \cos\beta.$$

However, because of the imperfect detectability of the sensor, the sensor fails to find some of them, and the measured area will be different from the nominal area n. Let P denote the detectability for this surface which we have computed in subsection 4.2. Then, the process of measuring the area by sampling n pixels can be modeled by a binomial distribution with mean nP and variance $nP(1-P)$. Assuming two standard deviations, the discrepancy in area measurement will be

$$\delta n = n - (nP - 2\sqrt{nP(1-P)}) = n(1-P) + 2\sqrt{nP(1-P)}.$$

Another uncertainty is also introduced in obtaining the surface orientation β from measured depths due to uncertainty in depth δz. If we estimate the surface orientation at a pixel by differentiating depths of neighboring pixels, then the uncertainty in surface orientation will be $\cos^2\beta \delta z$. However, since we have roughly n pixels in the region, the surface orientation will be averaged, reducing the uncertainty by a factor $\sqrt{n}$. Thus

$$\delta\beta = \frac{\cos^2\beta}{\sqrt{n}}\delta z$$

Finally, the estimation of area of the face, $A + \delta A$, is obtained by converting the image area into 3D space.

$$A + \delta A = \frac{n + \delta n}{\cos(\beta + \delta\beta)}$$

Thus, assuming that $\delta\beta$ is small, we see that

$$\delta A = A(1-P) + 2\sqrt{\frac{AP(1-P)}{\cos\beta}} + A\tan\beta \delta\beta$$

$$= A(1-P) + \sqrt{\frac{A}{\cos\beta}}(2\sqrt{P(1-P)} + \frac{\sin 2\beta}{2}\delta z)$$

In this way, we can predict what deviations from the nominal value of the area feature should be expected once we model the sensor and know its intrinsic detectability P and reliability δz.

6.3. Applying the Sensor Model to Aspect Structures

By using sensor model, we can predict the ranges of various feature values at each aspect. At each image, since a nominal value of a feature and its configuration with respect to sensor coordinates are given, we can predict the range of the feature value for each 2D face of the image by using the formula described above. Then, the range of the feature value at an aspect component is obtained as a sum of ranges of the feature values over its registered image components which can be reachable along *IS-AN-IMAGE-COMP-OF-ASPECT-OF+INV*. The predicted range will be stored in the slot of an aspect component frame.

Firgure 19 shows slots for this purpose. For example, area ranges, moment ranges, and moment ratio ranges are calculated at each image components, *IMAGE-COMP01*, *IMAGE-COMP12* which can be retrieved along the link stored in slot *IS-AN-IMAGE-COMP-OF-ASPECT-OF+INV* of *ASPECT-COMP10* frame in figure 14 (b). The sum of the ranges are stored in slot *AREA-VARIANCE*, *MOMENT-VARIANCE*, and *MOMENT-RATIO-VARIANCE* of *ASPECT-COMP10* frame. Similarly, feature ranges of aspect component relations, such as *DISTANCE-VARIANCE*, *MOMENT-ANGLE-P-TO-N-VARIANCE*, *SURFACE-ORIENTATION-ANGLE-VARIANCE*, are obtained and stored. These ranges of features will be retrieved by generation rules at compile time to generate an interpretation tree and by the execution process at run time in recognizing a scene.

7. Generating Programs

In this section, we will consider the final step of compilation of a recognition program: rule-based generation of a recognition strategy and conversion of the strategy into a executable program. As was in Section 2, the recognition strategy is represented by an interpretation tree which is made of two parts: the first part for classifying the input scene into one of the aspects and the second part for calculating the exact attitude.

7.1. Recognition Strategy: Classification

Strategy generation for aspect determination can be regarded as a process which classifies a group of aspect components into sub-groups of aspect components by applying classification rules recursively. At the beginning of the classification, a starting node is prepared, which contains all aspect components. We represent each classification stage as a node.

```
{{ ASPECT-COMP10

    ....
    (AREA-VARIANCE (13.94 14.85 15.75))
    (MOMENT-VARIANCE (22.77 25.06 27.34))
    (MOMENT-RATIO-VARIANCE (0.53 0.65 0.76))
    (VISIBLE-EDGE-LIST ASPECT-COMP10-VISIBLE-EDGE-LIST)

    ....
    }}

{{ ASPECT-COMP-RELATION-11-10

    ....
    (DISTANCE-VARIANCE (5.04 5.38 5.69))
    (MOMENT-ANGLE-P-TO-N-VARIANCE (1.29 1.53 1.8))
    (MOMENT-ANGLE-N-TO-P-VARIANCE NIL)
    (SURFACE-ORIENTATION-ANGLE-VARIANCE (0.04 0.21 0.40))

    ....
    }}
```

Figure 19:   Slots for representing uncertainty in features

The following sixteen rules have been prepared. Each rule tries to classify a group of aspect components at a node into smaller subgroups of aspect components by using the designated feature. For example, rule A1 will classify a group of aspect components comparing area sizes of their subaspect components.

A1: *face area*

A2: *face moment*

A3: *face moment ratio*

A4: *number of surrounding faces*

A5: *distances between surrounding faces and the face*

A6: *angles between moment direction of surrounding face and the face*

A7: *surface orientation differences between surrounding faces and the face*

A8: *face area of surrounding faces*

A9: *face moment of surrounding faces*

A10: *face moment ratio of surrounding faces*

A11: *surface characteristics of the face*

A12: *surface characteristics of surrounding faces*

A13: *surface characteristics distribution of the face*

A14: *surface characteristics distribution of surrounding faces*

A15: *edge distribution of the face*

A16: *edge distribution of surrounding faces*

The cost of calculations increases in order from A1 to A16; templates re required to calculate the features for the rules after A12. The order of preference in application is A1 to A16. Application of a rule proceeds in the following steps:

1. A rule selects a node which contains a group of aspect components.

2. It computes the threshold values of the feature to be used for classification from ranges of the feature values over the group of aspect components.

3. The rule classifies the group of aspect components into sub-groups by using the determined threshold values.

4. It generates new nodes for the newly generated subgroups of aspect components.

Since the preference of rules has been set in order of A1 to A16, a node will be kept divided by the applicable and the most preferable rules.

If no more rule is applicable (ie., no more nodes are dividable), application of rules A1 to A16 stops. Those nodes which contain only one aspect component are ready for the next stage of generating strategy for its attitude determination. At the termination, if there is a node which contains more than one aspect component and yet no rule is applicable to it, the parallel verification rule will be applied to the aspect components contained in the node. Since no further classification is possible, all possible aspect components in the node must be examined to see if any particular attitude is recognizable.

Once a tree is obtained by these rules, unnecessary branches are pruned. A rule may have generated a single child node from a parent node because the rule could not divide aspect components in the parent node. This rule-based generation of a strategy for classification has been implemented in OPS-5 [22].

In applying this method in practice, we require a principle to choose a particular object and thus a particular region in an input image from which to start a recognition process. For a bin-picking task, we assume that the highest object is the best object to recognize. Under this assumption, there are two alternatives for a starting region:

1. The largest region of the highest object (conservative principle)

2. Any region of the highest object (aggressive principle)

Since the conservative principle begins with a set of only the largest visible aspect components, one from each aspect, the interpretation tree will have a smaller number of nodes than the aggressive principle which will begins with a set of all aspect components. Therefore it will be more efficient in search than that for the aggressive principle, while it may be less reliable because the system may fail to find the largest region in the image.

7.2. Recognition Strategy: Attitude Determination

Once the aspect classification part of the interpretation is completed, the part for attitude determination is to be constructed next. This part is constructed for each aspect component of a leaf node to determine the precise attitude using the linear feature calculations. First the z axis direction of the object coordinate system is determined and then rotation angle around it is determined.

The following two rules are prepared for the determination of the z axis direction:

D1: *mass center of EGI distribution.*

D2: *extended Gaussian image.*

If there is no partial occlusion of visible faces over all possible attitudes within the aspect and all visible faces are planar surfaces, the EGI mass center by rule D1 is used to determine the viewer direction. In other cases, matching of EGI by rule D2 is used.

Once the viewer direction is determined, the rotation around the axis is obtained next. One of the following six rules will be adopted by examining by one to see if it constrains the freedom of rotation:

R1: *position of detectable region distribution.*

R2: *position of EGI distribution*

R3: *moment direction*

R4: *EGI moment direction*

R5: *position of the surface characteristics distribution*

R6: *position of the edges.*

7.3. Executable Program

Once recognition strategy has been obtained with the necessary rules to be used at each stage, we have to convert the recognition strategy into an executable program. We are using the technique of object-oriented programming, because it simplifies to combine various elementary modules into a complete program.

Each node of the tree is converted into an "object" in object-oriented programming. We are preparing a library of object prototypes which will be used to execute matching operations between image regions and models according to rules [17]. Each rule has one corresponding prototype in the library. Right now, we are working on an efficient way to organize the library. A necessary instance of prototype (ie., object) to be adopted at a node is generated from the corresponding rule of the node. The descendant nodes which will receive a message from this node are inserted in slot *EXECUTION-DESTINATIONS* of the object. Slot *EXECUTION-ARGUMENTS* contains the threshold value and other matching templates. Actual

operations are executed as message passing between objects (nodes). The operation begins by sending an execution message and a target region to the starting node object. After that event, a chain of operations takes place by passing execution messages from object to object. When an object receives an execution message, the object executes a matching method which had been particularly adopted to the node. Since regions in the image are also implemented as objects, a message is sent to the target region to receive a necessary feature value from it.[10] Then, the matching method compares the value which is returned from the target region with the values in *EXECUTION-ARGUMENTS* slot. Based on the comparison result, the object determines to which object in *EXECUTION-DESTINATION* slot it should send an execution message next. This event is repeated until an execution message reaches one of the leaf objects of the tree. At that point, the tree determines the object attitude exactly.

Rule-based automatic generation of an interpretation tree has been applied to an object shown in Figure 20(a), which has fourteen aspects as shown in Figure 20(d)[11]. The aggressive principle was chosen to select the starting region. The generation process generated the interpretation tree shown in Figure 20(b). After the pruning operation, the result was an interpretation tree shown in Figure 20(e). This pruning operation reduced the depth of the interpretation tree from 14 levels to 4. The obtained recognition strategy is converted into a recognition program by using the object library (See Figrue 20(f)).

The generated program is applied to the scene as shown in Figure 21(a). Figure 21(b) shows the needle map, Figure 21(c) shows the segmented regions based on surface orientation distribution, and Figure 21(d) shows edge distributions superimposed on the region distributions. The highest region, determined by the dual photometric stereo (indicated by an arrow in Figure 21(c)), is given to the program. The black nodes in Figure 21(e) indicates the nodes which receive the execution messages in the real run. The program classifies the region to the corresponding aspect successfully.

---

[10]This mechanism is particularly useful when calculation of a feature is expensive, such as region relation. The system also converts an image value into a model value by using this mechanism. See [17] for more details.

[11]In this experiment, we only consider the northern hemisphere of the Gaussian sphere as viewer directions for the sake of simplicity. See Figure 20(c)
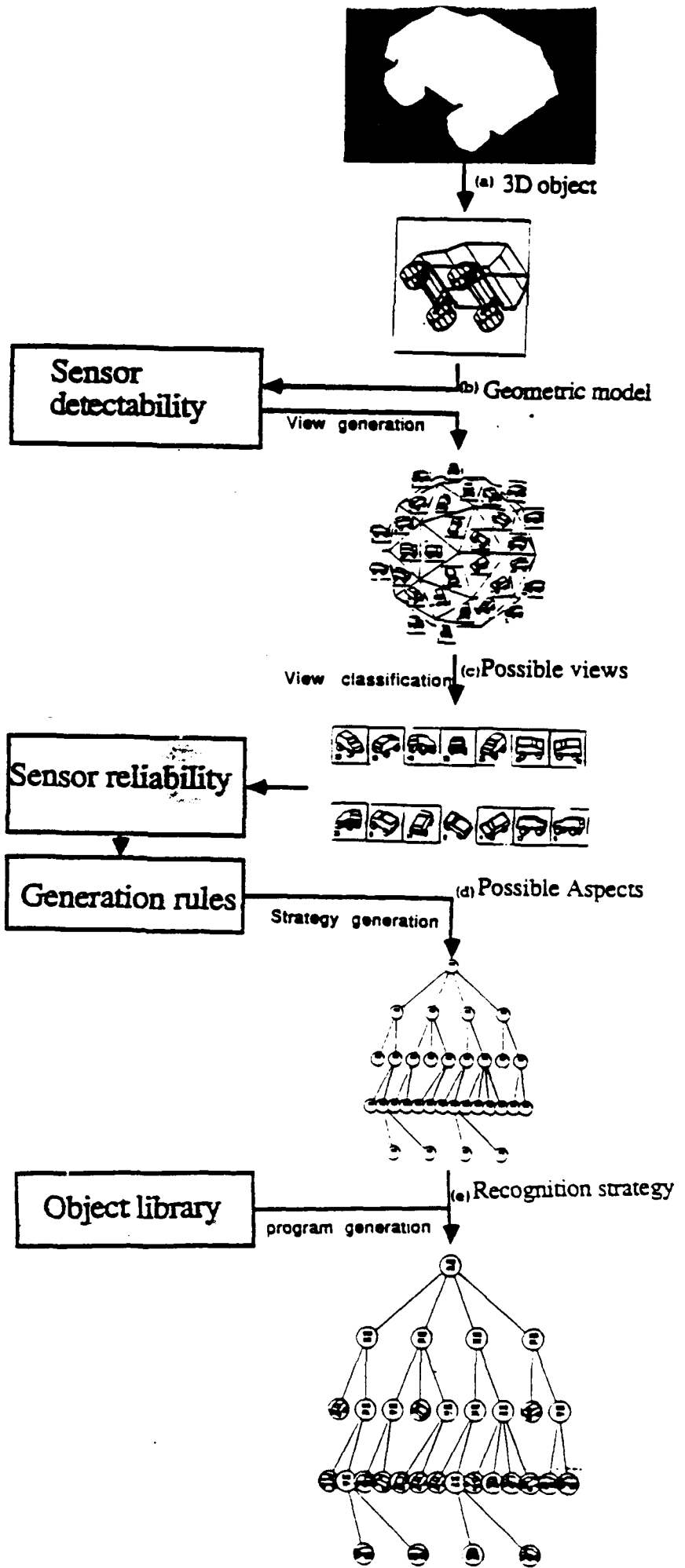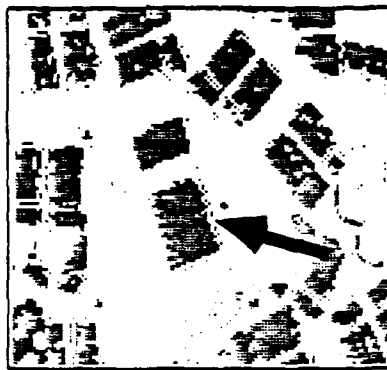
(a) 3D object

(b) Geometric model

Sensor detectability

View generation

(c) Possible views

View classification

Sensor reliability

(d) Possible Aspects

Generation rules

Strategy generation

(e) Recognition strategy

Object library

program generation

Figure 20: Generation of an interpretation tree for a toy wagon:

(f) Recognition program

165
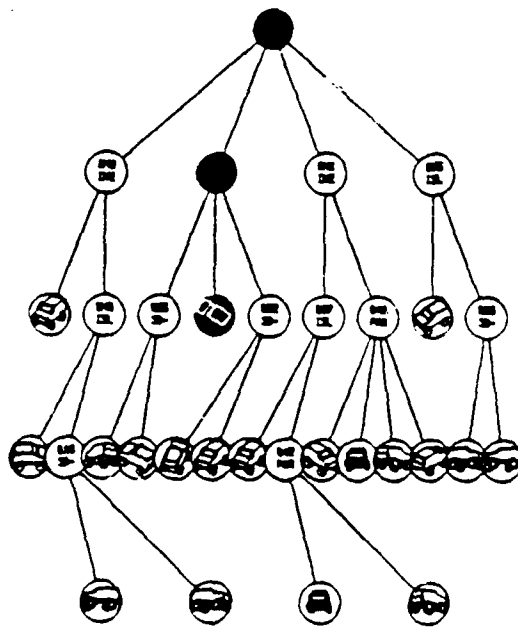
Figure 21: Tree execution: (a) Input scene; (b) Surface orientation distribution of the scene; (c) Segmented regions using shadows and surface orientation discontinuities. The arrow indicates the target region selected by the conservative strategy; (d) Edge distributions superimposed on the region map; (e) Execution result. The target region is classified into the corresponding aspect successfully.

166

8. Future Directions

This paper has discussed issues and techniques to automatically compile object and sensor models into a visual recognition program. This automatic generation requires several key components: object modeling, sensor modeling, strategy generation, and program generation. Especially we have argued for the importance of sensor modeling, as it has been studied very little in the past. We have presented our effort toward a systematic way to modeling sensors: representation of geometrical relationships between a sensor and an object feature and calculation of a feature's detectability and reliability. Actual creation and execution of interpretation trees by our method has been demonstrated.

Vision has been recognized as an important, versatile sense for industrial applications. Yet, the number of successful applications seems to be far below the expectation. Apart from the large computational requirement and the cost, one of the serious factors which hinder wider application of vision is the time and expertise required to program a vision system. The automatic generation of recognition programs by compiling object and sensor models will mend the situation.

Moreover, automatic program generation may open a new dimension of capability in model-based vision when it comes to special sensors such as synthetic aperture raaars (SAR) or FLIR. In those cases, since their sensor characteristics are not very intuitive, even capable vision implementors may not be doing the best job and an automatic and mechanical method of generating programs may be more advantageous.

Another area of investigation is learning from real scenes. For example, the range of a feature value is currently obtained solely from the analysis of sensor reliability and detectability. This information can be learned and modified by running the interpretation tree first generated from automatic analysis. The parameters used at branches are improved iteratively through real execution. Furthermore, branching structures themselves can be modified slightly. A critical difference of this approach from usual learning of recognition algorithms from scratch is that we start with the "skeleton" strategy which is more or less valid. Therefore there is a good chance that the final algorithm is truly competent.

Acknowledgement

References

[1]   Agin, G.J. and Binford, T.O.
        Computer description of curved objects.
        In *International Joint Conf. on Artificial Intelligence*, pages 629-640. Stanford, CA,
            August, 1973.

[2]   Baker, H.H. and Binford, T.O.
        Depth from edges and intensity based stereo.
        In *International Joint Conf. on Artificial Intelligence*. 1981.

[3]   Ballard, D.H.
        Generalizing the Hough transform to detect arbitrary shapes.
        Pattern Recognition 13(2), 1976.

[4]   Barrow, H.G. and Popplestone, R.J.
        Relational description in picture processing.
        In Meltzer, B. and Michie, D. (editor), *Machine Intelligence 6*. Edinburgh University
            Press, Edinburgh, Scotland, 1970.

[5]   Besl, P.J., and Jain, R.J.
        Intrinsic and extrinsic surface characteristics.
        In *CVPR*. IEEE computer society, San Francisco, 1985.

[6]   Binford, T.O.
        Visual perception by computer.
        In *Proc. IEEE Systems Science and Cybernetics Conf.*. IEEE, 1971.

[7]   Binford, T.O.
        Survey of model-based image analysis systems.
        The International Journal of Robotics Research 1(1), 1981.

[8]   Bolles, R.C. and Horaud, P.
        3DPO: A three-dimensional part orientation system.
        In Kanade, T. (editor), *Three-Dimensional Machine Vision*. Kluwer, Boston MA, 1987.

[9]   Bolles, R. and Cain, R. A.
        Recognizing and locating partially visible objects: the local-feature-focus method .
        The International Journal on Robotics Research 1(3), 1982.

[10]   Brady, J.M. and Asada, H.
        Smoothed local symmetries and their implementation.
        The International Journal of Robotics Research 3(3), 1986.

[11]   Brady, M., Ponce, J., Yuille, A., and Asada, H.
        Describing surfaces.
        In Hanafusa, H. and Inoue, H. (editors), *Proc. 2nd International Symposium on Robotics
            Research*. MIT Press, Cambridge, MA, 1985.

[12]   Brooks, R.A.
        Symbolic reasoning among 3-D models and 2-D images.
        Artificial Intelligence 17(1-3), 1981.

[13]   Brou, P.
         Using the Gaussian image to find the orientation of object.
         The International Journal of Robotics Research 3(4), 1983.

[14]   C. Brown.
         Fast display of well-tesselated surfaces.
         Computer and Graphics 4(4):77-85, April, 1979.

[15]   Canny, J.F.
         Finding edges and lines in images.
         Technical Report AI-TR-720, Massachusetts Institute of Technology, Artificial
                 Intelligence Laboratory, 1983.

[16]   Chakravarty, I. and Freeman, H.
         Characteristic views as a basis for three-dimensional object recognition.
         In *Proc. The Society for Photo-Optical Instrumentation Engineers Conference on Robot
                 Vision.* SPIE, Bellingham, Wash., 1982.

[17]   Chang, H., Ikeuchi, K., and Kanade, T.
         Model-based vision system by object-oriented programming.
         Technical Report CMU-RI-TR-88-3, Carnegie Mellon University, The Robotics Institute,
                 March, 1988.

[18]   Chin, R.T. and Dyer, C.R.
         Model-based recognition in robot vision.
         ACM Computing Surveys 18(1), March, 1986.

[19]   Cutrona, L.J.
         Synthetic Aperture Radar.
         In Skolnik, M.I. (editor), *Radar Handbook*, chapter 23. McGraw Hill, New York, 1970.

[20]   Falk, G.
         Interpretation of imperfect line data as a three-dimensional scene.
         Artificial Intelligence 3(2), 1972.

[21]   Faugeras, O.D. and Hebert, M.
         The representation, recognition, and locating of 3-D objects.
         The International Journal of Robotics Research 5(3), 1986.

[22]   Forgy, C.L.
         OPS5 User's manual.
         Technical Report CMU-CS-81-135, Carnegie Mellon University, Computer Science
                 Department, July, 1981.

[23]   Goad, C.
         Special purpose automatic programming for 3D model-based vision.
         In *Proc. of DARPA Image Understanding Workshop.* DARPA, 1983.

[24]   Grimson, W.E.L.
         From Images to Surfaces: a computational study of the human early visual system.
         MIT Press, Cambridge, MA, 1981.

[25]   Grimson, W. E. L. and Lozano-Perez, T.
         Model-based recognition and localization from sparse range or tactile data.
         The International Journal of Robotics Research 3(3), 1984.

[26] Hebert, M. and Kanade, T.
The 3D Profile method for object recognition.
In *CVPR*. IEEE computer society, San Francisco, June, 1985.

[27] Hebert, M. and Kanade, T.
Outdoor scene analysis using range data.
In *Proc. of Intern. Conf. on Robotics and Automation*, pages 1426-1432. IEEE Computer
Society, San Francisco, April, 1986.

[28] Herman, M.
Matching three-dimensional symbolic descriptions obtained from multiple views.
In *CVPR*. IEEE computer society, San Francisco, June, 1985.

[29] Horn, B.K.P.
Obaining Shape from Shading.
In Winston, P.H. (editor), *The Psychology of Computer Vision*. McGraw-Hill, New York,
1975.

[30] Horn, B.K.P.
Extended Gaussian Images.
Proc of the IEEE 72(12), December, 1984.

[31] Ikeuchi, K.
Recognition of 3-D objects using the extended Gaussian image.
In *International Joint Conf. on Artificial Intelligence*. 1981.

[32] Ikeuchi, K.
Determining attitude of object from needle map using extended Gaussian image.
Technical Report AI memo No. 714, MIT Artificial Intelligence Laboratory , Cambridge,
MA, 1983.

[33] Ikeuchi, K.
Determining a depth map using a dual photometric stereo.
The International Journal of Robotics Research 6(1), 1987.

[34] Ikeuchi, K.
Generating an Interpretation Tree from a CAD Model for 3-D Object Recognition in Bin-
Picking Tasks.
International Journal of Computer Vision 1(2), 1987.

[35] Ikeuchi, K., Nishihara, H.K., Horn, B.K.P., Sobalvarro, P., and Nagata, S.
Determining grasp points using photometric stereo and the PRISM binocular stereo
system.
The International Journal of Robotics Research 5(1), 1986.

[36] Ikeuchi, K. and Horn, B.K.P.
Numerical shape from shading and occluding boundaries.
In Brady, M.J. (editor), *Computer Vision*. North-Holland, Amsterdam, 1981.

[37] Ikeuchi, K. and Kanade, T.
Modeling sensor detectability and reliability in the sensor configuration space.
Technical Report CMU-CS-87-144, Carnegie-Mellon University, Computer Science
Department, 1987.

[38] Jarvis, R.A.
A laser time-of-flight range scanner for robotic vision.
IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-5(5), 1983.

[39] Koenderink, J. J. and Van Doorn, A. J.
Geometry of binocular vision and a model for stereopsis.
Biological Cybernetics 21(1), 1976.

[40] Koenderink, J. J. and Van Doorn, A. J.
Internal representation of solid shape with respect to vision.
Biological Cybernetics 32(4), 1979.

[41] Koezuka, T., and Kanade, T.
A technique of pre-comiling relationship between lines for 3D object recognition.
In *Proc. Intern. Workshop on Industrial Applications of Machine Vision and Machine Intelligence*. IEEE Industrial Electronics Society, February, 1987.

[42] Koshikawa, K.
A polarimetric approach to shape understanding of glossy objects.
In *Proc. of 6th Intern. Joint Conf. on Artificial Intelligence.* 1979.

[43] Koshikawa, K. and Shirai, Y.
A Model-based recognition of glossy objects using their polarizational properties.
Journal of Robotics Society of Japan 3(1), 1985.
(in Japanese, brief English version is avaiable as Y.Shirai, Three-Dimensional Computer vision, pp.153-156, pp.274-276, Springer-Verlag, 1987, Berlin).

[44] Little, J.J.
Determining object attitude from extended Gaussian images.
In *Proc. of 9th Intern. Joint Conf. on Artificial Intelligence.* 1985.

[45] Marr, D. and Hildreth, E.
Theory of edge detection.
Proc. of the Royal Society of London B 207, 1980.

[46] Marr, D. and Poggio. T.
A computational theory of human stereo vision.
Proc. of the Royal Society of London B 204, 1979.

[47] McKeown, D.M., Harvey, W.A., and McDermott, J.
Rule based interpretation of aerial imagery.
IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-7(5), 1985.

[48] Mensa, D.L.
High Resolution Radar Imaging.
Artech House, Dedham MA, 1981.

[49] Milenkovic, V.J. and Kanade, T.
Trinocular vision: using photometric and edge orientation constraints.
In *Proc. of DARPA Image Understanding Workshop*. DARPA, Miami Beach, FL, December, 1985.

[50] Miwa, H., and Kanade, T.
On line extraction.
Technical Report, Carnegie Mellon University, Robotics Institute, Pittsburgh, PA, 1987.
in preparation.

[51] Nagao, M. and Matsuyama, T.
A structural analysis of complex aerial photograph.
Plenum, 1980.

[52] Ohta, Y. and Kanade, T.
Stereo by intra- and inter-scanline search using dynamic programming.
IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-7(2), 1985.

[53] Oshima, M. and Shirai, Y.
Object recognition using three-dimensional information.
IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-5(4), July, 1983.

[54] Oshima, M. and Shirai, Y.
A model based vision for scenes with stacked polyhedra using 3D data .
In *Proc. Intern. Conf. on Advanced Robot (ICAR85)*. Robotics Society of Japan, 1985.

[55] Pentland, A. P.
Perceptual Organization and the Representation of Natural Form.
Artificial Intelligence 28(2), 1986.

[56] Perkins, W. A. .
Model-based vision system for scene containing multiple parts.
In *Proc. 5th International Joint Conference on Artificial Intelligence*. 1977.

[57] Plantinga, H. and Dyer, C.
The APS: a continous viewer-centered representation for 3D object recongition.
Technical Report 682, University of Wisonsin-Madison, Computer Science Department,
     1987.

[58] Roberts, L.G.
Machine perception of three-dimensional solids.
In Tipplett, J.T. (editor), *Optical and Electro-Optical Information Processing*. MIT Press,
     Cambridge, MA, 1965.

[59] Shafer, S. A. and Kanade, T.
*The Theory of Straight Homogeneous Generalized Cylinders, and A Taxonomy of
     Generalized Cylinders.*
Technical Report CMU-CS-83-105, Carnegie-Mellon University, Computer Science
     Department, January, 1983.
(also a shorter version is presented at *Proc. DARPA Image Understanding Workshop*.
     pp. 210-218, Washington, D.C., June 1983.).

[60] Sugihara, K.
Automatic construction of junction dictionaries and their exploitation for analysis for
     range data.
In *Proc. of 6th Intern. Joint Conf. on Artificial Intelligence*. 1979.

[61]   Thorpe, T., Hebert, M., Kanade, T. and Shafer, S.
        Vision and Navigation for the Carnegie-Mellon NAVLAB.
        In Traub, J. (editor), *Annual Reviews of Computer Science*, pages 521-556. Annual
            Reviews Inc, 1987.

[62]   Thorpe, C., and Shafer, S. A.
        Correspondence in Line Drawings of Multiple View⊏.
        In *Proc. of 8th Intern. Joint Conf. on Artificial intellignece.* 1983.

[63]   Tomiyasu, K.
        Tutorial review of Synthetic-Aperture Rader(SAR) with applications to imaging of the
            ocean surface.
        Proc. of the IEEE 66(5), May, 1978.

[64]   Woodham, R.J.
        Reflectance Map Techniques for Analyzing Surface Defects in Metal Castings.
        Technical Report AI-TR-457, Massachusetts Institute of Technology, Artificial
            Intelligence Laboratory, Cambridge, MA, 1978.

[65]   Yoda, H., Motoike, J., and Ejiri, M.
        Direction Coding Method and its Application to Scene Analysis.
        In *Proc. 4th Joint Conference on Artificial Intelligence.* International Joint Conf. on
            Artificial Intelligence, 1975.

SPACE-VARIANT VISION: IMPLEMENTATION OF SCANPATH AND BLENDING ALGORITHMS FOR CONTOUR-BASED SCENES

Yehezkel Yeshurun (1)

Eric L. Schwartz (2)

1 Department of Computer Science
Sackler faculty of Exact Science
Tel Aviv University
Ramat Aviv, Israel 69978

2 Computational Neuroscience Laboratories
Courant Institute
Department of Computer Science
715 Broadway
New York, N.Y. 10003

Introduction

When we view a scene, we have the subjective impression that what we see is stable and constant, both in position and resolution. However, it is not hard to show that this impression is far from correct. For example, if we try to read a newspaper that is slightly off-center (see Fig. 1) , we become aware that the very high resolution provided in the region of our fixation (foveal projection) falls off rapidly toward the edges of our field of vision. The fact that the human visual representation is strongly space variant, implies that the human system builds up a representation of a scene through multiple fixations during scanning.

The space variant nature of the human visual system is well understood, at least to the level of primary visual cortex. The threshold for visual acuity, stereo acuity, motion, and other psychophysical quantities scale at least roughly as the inverse of dis-

tance from the fovea. There is general consensus[1, 2, 3] that the spatial representation of the visual field [1] , at the level of the primary visual cortex, is approximated by a complex logarithmic mapping[4]. Figure 1 and Figure 6 of this paper show natural scenes processed by this form of mapping function. We are thus in a position to provide realistic estimates of the nature of a specific space variant imaging system: that of the human.

In the present paper, we discuss three algorithms related to the "blending" of a single scene from multiple frames acquired from a space variant sensor. We used contour based scenes, rather than gray scale scenes, in order to focus attention on the problem space variance, as opposed to segmentation. The following generic problems are raised by considering a a space variant system:

1.) Given a series of space-variant contour based scenes, with different "fixation points", how might one fuse these into a single, multi-scan view, which incorporates the information present in the individual scans?

2.) How might one choose successive fixations points, in order to rapidly gather shape dependent data? Is there a simple attentional algorithm for contour based scenes?

3.) How could one quantify the rate of convergence of such a system, as a function of the number of scans? What is the rate of convergence suggested by such a metric? [2]

In the present work, we do not address the classical issues of how the system (human or machine) is to obtain knowledge of its motor state (see 5). Our intention here is to discuss the image processing problem of blending together multiple scans, obtained from a strongly space variant sensor, and the problem of choosing a "scan

---

[1] In this paper, we do not discuss the detailed spatial architecture of primary visual cortex, which would include details such as ocular dominance columns, orientation columns, etc. We are only concerned here with the first order topographic structure of the human visual system, as a model for space variant machine vision systems.

[2] In addition to these purely computational issues, the human system has also needed to: 1.) evolve systems of accurate motor control, 2.) provide information to the organism about the current motor state (i.e. direction of gaze). This aspect of the problem has been much discussed under the terms proprioceptive perception, efference copy, corallary discharge, etc.[5].

path" which provides optimal information about the scene.

Another interpretation of the work described here might be made, entirely within the context of machine vision. Assuming that a space variant sensor similar to a human retina were available, it would be necessary to consider some of the issues discussed in the present paper: how should one choose a series of fixation points for such a sensor, how would one blend the successive frames, and how could one place a metric on the quality of this scanning process?

The space-variant image and boundary-angle function

We define the resolution at the point v of an image as the function $R_p(v)$, where p is the spatial location of a fixation point and R is a monotonic non-increasing function of $|v-p|$. This is to say that R is proportional to the reciprocal of the minimal distinguishable distance (i.e visual acuity). In the current context the exact specification of R is not crucial; any R having the mentioned attribute can be used. The following discussion uses a function of the form $\frac{c}{|v-p|}$, for $v \neq p$, where c is a constant.

This definition might be applied to any gray-scale image (see Fig. 1). In the current application we consider only contour based images. This situation can arise either naturally, when a scene is two-dimensional and consists only of contours, or artificially, after an edge-detection mechanism has been applied to an image of a complex three-dimensional scene (segmentation).

Boundary contour descriptor

In applications in which a one-dimensional representation of contours is desired, it is customary to use the boundary-angle function $\theta(l)$, which is the angle of the tangent to the contour, as a function of the arc-length unit l. In the current application, since we have discrete points connected by line segments (i.e polygons) , we use the representation $\Theta(l)$, which is the difference between two consecutive angles of the polygon. This one-dimensional representation of contours is most useful in shape-

recognition tasks, where it is further processed by a Fourier transform to yield the Fourier descriptors (FDs) of the contour [6]. There are also some indications that the FD of a shape might be useful as a shape descriptor in physiological studies of the primate visual system[7].

We apply, spatial-variant resolution to both the image of the contour in the x/y plane and to the boundary-angle $\Theta(l)$ representation of it, as explained below (see also Figure 2).

1) The original contour is represented by line segments between the points $\{U_i, i=1,k\}$. We assume that the distance between these points represents the highest possible resolution of the "viewer."

2) A new contour is defined by a fixation point: Given a fixation point p, and a contour point U_i, the value of $R_p(U_i)$ determines the next point U_j. Thus, starting at U_0, this procedure yields a contour whose points are a subset of the original points.

3) The boundary angle of the new contour, $\Theta_p(U_i)$, $i\varepsilon\{1,k\}$, is obtained. To allow reconstruction of the original image, we also keep the resolution value $R_p(U_i)$ for each U_i.

In the x/y plane, variable resolution produces a detailed image near the fixation point and a "blurred" image away from the fixation point. In the boundary-angle representation, the neighborhood of the fixation point is properly described, while other areas retain only smoothed, low-frequency details. The parameters used in this work yield a ratio of 1:10 between the full resolution image and a single space-variant view, which is in good agreement with the functional form of human visual acuity[3].

---

[3] One recent estimate of primate magnification factor[1] suggests that there is a 10:1 decrease in spatial resolution of a stimulus between the fovea and five degrees of eccentricity. This is a reasonable "viewing aperture" for shape perception. Note that a 10:1 (linear) change corresponds to a 100:1 area change, and that this area change is a more relevant index of "data compression".

blending boundary-angle functions and images

For a given fixation point, there exists a corresponding representation of the original contour. Several fixation points $\{p=p_1 \cdots p_n\}$ produce different representations of the same contour. This situation is shown in Figure 3, in which images are viewed from several different points. Although the boundary-angle function $\Theta_p(U_i)$ is quite detailed near the corresponding fixation point, it just roughly approximates the original boundary-angle function in all the other areas.

Because resolution depends only on the distance between a given point and the fixation point, and because the most detailed boundary functions (or images) are obtained for high-resolution areas, an appropriate blending scheme should use the "best" of each view. The only information the blending scheme needs is the resolution associated with each point in the subcontour, which is kept when the subcontour is calculated. Thus, the reconstructed boundary-angle function is

$$\Theta^*(U_i) = \Theta_f(U_i)$$

such that

$$R_f(U_i) = max_{p=p_1 \cdots p_n} \{R_p(U_i)\}.$$

The reconstructed function $\Theta^*(i)$ is an approximation to the original $\Theta(i)$. This approximation depends on the number of fixation points and their location. A more elaborate blending scheme might also depend on the "scanpath" or sequence of fixation points humans select when viewing a given scene[8].

Choice of scan path: an "attentional" algorithm

Although early vision and artificial intelligence (late vision?) have received a great deal of attention recently, a great intermediate area exists which has received little study in this context, and that is the subject of "attention" itself. A single scan provides partial information about a scene. Assuming that a unified representation of

the scene can be extracted from successive scan, we must address the problem of locating the fixation points,in such a way as to provide maximal information to the imaging system. This represents an ill defined problem, as difficult issues relating to context and goal direction are implied by it. However, little advantage can be gained from a space variant system without providing an attentional algorithm. In the following, we will discuss a simple candidate for attentional choice of successive fixation points.

In psychophysical contexts, the nature of visual scanning has been extensively explored (e.g., 9). In general, fixation points tend to cluster around sharp edges, ends of lines, and locations where some "unpredictable" change takes place. Although most existing research considers only the question of the location of the fixation points, some of the literature does pay attention to the temporal ordering of these points, which is termed the "scanpath"[8].

In our case, the scene consists of contours. The curvature of the contours is very likely to be a prime fixation-point "attractor", since large curvature represents rapid rate of change of boundary orientation. We can represent the curvature in terms of a boundary-angle function, indicating areas of high curvature by corresponding peaks in the function. A simple form of attentional algorithm, then, consists of the following steps:

1) Chose (randomly, or by any method) an initial fixation point.

2) Calculate the boundary-angle function according to the current fixation point.

3) Select the next fixation point according to the maximum of the boundary-angle function $\Theta_p(U_i)$.

4) Keep the boundary angle function and the corresponding resolution values. Keep a reference point in the current fixation, that will be associated with a point in the next fixation.

5) Blend the views and the boundary angle functions to yield a single view/function.

6) Go to step 2, until "convergence" (see below).

Such a procedure is shown in Figure 4. The fixation points in this figure seem plausible in comparison with the points that one would likely select without using the algorithm. However, the algorithm has one drawback. In cases where several high values of the boundary-angle function cluster together, the algorithm picks several fixation points at almost the same place. Because the scans obtained from adjacent fixation points do not differ much, and because the foveal area can cover several points of high curvature, this clustering of points is redundant.

In order to remove the redundancy, we modify the algorithm (in step 3) by considering $\Theta(U_j)W(U_j)$ instead of $\Theta(U_j)$. The weight function $W(U_j)$ can be used to enhance (or mask) selected features. If W is chosen such that it equals 1 everywhere except for a neighborhood of the fixation point where it vanishes, the redundancy problem is solved. In other words, after a fixation point is selected, the relevant foveal area (i.e., the area immediately surrounding the fixation point, where the high resolution still holds) is not counted when the algorithm searches for the next-higher value. Figure 4b shows the results of this approach.

One might also select W to be $\frac{1}{R}$, thus emphasizing "remote" features rather than "close" ones. Finally, W might contain some random fluctuations, in order to avoid the possibility of being "trapped" between two features.

The algorithm needs a reference point that is shared between each two succesive fixations: this is necessary when the views, or the boundary angle functions, are "tailored" together.

181

Convergence and norms

Because our figures consist of simple contour drawings, it is easy to define a norm that compares composite space-variant scenes after n scans with the original high-resolution scene. A reasonable choice for this norm is a least-squares measure of the two boundary-angle functions. Thus, let Δ_n represent the difference between the full-resolution scene and the composite scene after the incorporation of the nth fixation point : $\Delta_n = |U - C_n|$.

Using this norm, it is possible to define the convergence rate as a function of the scanpath. Thus, for a sequence of fixation points $p_1, p_2 ... p_n$, we define the rate of convergence for the scan path at point n, as $\Delta_n - \Delta_{n-1}$. This method is suitable for the purpose of the algorithms evaluation or for calibration, when we have access to the full resolution contour. However, in a "real-time" situation (i.e in robotic vision), the full resolution image is not necessarily available. Thus, we can define Δ_n as $|C_n - C_{n-1}|$, and base the "convergence" decision on it (see Fig. 5). If one thinks of n as a time variable then this measure indicates the "rate" of error-reduction.

Thus, one algorithm for adding scanpaths might be based on the addition of a new point which, among all the possible fixation points, maximizes the above "rate" of convergence. Conversely, the addition of new points becomes unecessary when no points can be found that significantly improve the rate of convergence. The algorithm we propose rapdily converges: it is monotonic, in the sense that only "better" resolution points are introduced, and it is bounded by the original set of points which constitutes the object. Figure 5 shows an example of an aircraft silhouette which is scanned by this algorithm, with a plot of convergence based on the latter method described above. It is clear that there is rapid convergence to an accurate representation of the boundary of the figure. It is interesting to note that [8] report that humans typically view scenes with perhaps 3 - 8 scans; our algorithm also converges quite rapidly, in this case in which parameters of space variance derived from human vision have been

In more general cases, however, the choice of a norm is likely to be quite difficult. In the general case, both the attentional algorithm and the norm used to evaluate its success would likely be dependent on past experience, the goal-directed state of the imaging entity, and the full context of the current task. In lieu of engaging in this full-blown algorithmic study of visual attention, we propose that the simple curvature based norm and scanning algorithm outlined above provides an initial step in the direction of understanding visual attention, and is one which is optimal in those situations in which a value-neutral estimate of boundary curvature is the desired information.

Implication of space variant image processing to gray-level images.

Though we address mainly contour-based images in this work, it might be of interest to point out its application to gray-level images, especially from the aspect of "data compression".

The human visual field subtends roughly 100 x 100 degrees[10] , with a maximum resolution of about 1 minute of arc (foveal). Using a space invariant sensor (e.g. conventional CCD camera), one would have to resolve 6000x6000 samples (1 minute of arc x 100 degrees in each direction). In order to achieve this performance, one would have to sample at 2-3 times this resolution, in each dimension. An image of 16000x16000 would provide this performance, but would extend close to the giga-pixel range in size.

We have experimentally demonstrated this estimate by digitizing[4] a conventional eye-chart, at a distance of 20 feet, using a wide angle (fisheye) lens, which recorded from about 80 degrees of field. Figure 6 shows the "full scene", and a highly

---

[4] We used a conventional NTSC frame grabber, at 480x525 resolution, together with a polar coordinate mosaic technique[11] to produce this simulation.

magnified detail of the eye-chart, at the center. We continued to magnify the scene, until the 20/20 line of the eye-chart was visible (indicating a resolution of about 1 minute/arc). We calculate that this occured at an effective sampling resolution of 16,0000x16,0000 pixels.

Although both of the previous estimates are ad-hoc, they agree well enough to suggest that the effective resolution of a single scan of the human system is equivalent, were it recorded by a space invariant system, to a 1/4 giga-pixel image. Now, this estimate of 1/4 giga-pixel is based on the use of a constant resolution system, which extended over 100x100 degrees, at full visual acuity. In fact, we simulated the logarithmic structure of the human visual system, and our simulated image occupied only about 16000 pixels (see figure 6). Naturally, we only obtained high resolution over a small "foveal" representation with this simulation; in order to use this approach effectively, multiple scans would need to be performed. However, with a relative data compression of about 16,000 : 1 , we can afford to perform the scanning process over a number of fixation points. Even 16 sucessive fixations would yield an effective 1000:1 data compression relative to a constant resolution system, provided that one obtained a satisfactory representation of the image regions of interest.

Summary

Space variant imaging has been little explored in the context of machine vision, but is a major area of interest in the context of biological vision. Space variant imaging provides a number of advantages, and difficulties, with respect to conventional space invariant systems. One advantage is that very large fields of view can be covered, and very high resolution can also be provided.This leads to a form of image data compression which can be extremely large. However, a number of algorithmic difficulties are introduced by considering strongly space variant systems. Attentional algorithms are required to make effective use of the small high resolution "fovea", while other algorithms are required to "fuse" successive space variant scans.

In the present paper, we have provided preliminary solutions to each of these issues. Using our algorithms, we obtain satisfactory convergence, for reasonable parameters of space variance derived from human vision, over a small number of scans (perhaps 3-5 scans).

The possibility that space variant sensors (e.g. CCD's) may become available for application in machine and robotic vision should provide some motivation to begin studying the issues which such a sensor would provide. Perhaps the possibility that some of the high performance of the human visual system derives from its use of a space variant architecture may provide some impetus to develop such a sensor.

Figure 1. Figure 1A simulates six successive scans of a newspaper, using a cortical map function derived from primate data[6] ,a reading distance of about twenty centimeters, and about 1.5 degrees of visual field on each side of the fixation point. Each of the small "bow ties" represents the cortical "image" of a section of newspaper print. Thus, the first frame is fixated on the letter "o" in the word "roaches". There are two "bow ties" representing the left and right visual fields. The newspaper is then scanned, and the corresponding cortical "images" are presented in the figure. Note the strong space variance, even for the central few degrees of visual field.

Figure 1B shows these six scans projected back to the visual field, and "fused" into a single scene[13]. The region of text scanned, which read " roaches don't die..", and too some extent the lines above and below this line, are seen clearly, but there is a rapid loss of detail in the text regions which are not close to the scanned text. Figure 6 of this paper shows a wide angle simulation of the human visual field and cortical image.

Figure 2. A: Images (left) and their boundary-angle functions (right). Top: the original contour (black silhouette) and its boundary-angle function. Bottom: the image as it is "viewed" from the fixation point (indicated by a star), with space-variant resolution. The tail of the airplane, being fairly far from the fixation point, is described very roughly. Therefore, the boundary-angle function bears only a rough resemblance to the original function.

B: A scene consisting of several planes silhouetes (a), as it is "received" from different fixation points (b-d). The fixation points are depicted by an asterix. The original airplane silhouette consists of 243 points, and the space-variant silhouettes average 5 points (for the less detailed ones) to 40 points (for the highly detailed).

Figure 3. A: View of a triangle from three fixation points. The contour of the original

triangle (top) is seen from three fixation points, each in the neighborhood of a particular vertex. These views are indicated by the corresponding boundary-angle functions. For each fixation point, only the closest vertex and its neighborhood are detailed, while the other vertices are approximated roughly. The reconstructed boundary-angle function (bottom) consists of the "best" contribution from each space-variant view.

B: a silhouette of an airplane, viewed from three fixation points, selected (by hand) because they are near areas containing many details. Details as in A.

Figure 4. A: Images (left) and the corresponding boundary-angle functions (right). The top row shows the original image and function; the next three rows represent three fixation points (denoted by small stars on the images), and the bottom row shows the integrated image and function. The fixation points, which are selected automatically, are the spatial locations that correspond to the three largest values of the original boundary-angle function (denoted by bars under the function).

B: Results of the modified algorithm. The fixation points are chosen by the maximum of $\frac{\Theta(U_i)}{R(U_i)}$.

Figure 5. Converging rate of the algorithm, as depicted by the difference Δ_n between successive blended figures. Left: blended figures after 1,2,3..8 fixation points. Right: Δ_n versus number of fixation points. Δ_n is the mean square error between two succesive figures, and is normalized to [0,1].

Figure 6. Figure 6A shows a wide angle fish eye view of a scene in the hall of our laboratory. A ladder is to the right, an eye chart is in the very center of the frame (almost invisible). The original version of this scene was digitized to an effective resolution of 16000x16000 pixels by a polar coordinate mosaic technique. A "blow-up" of the central region of this original frame is shown in figure 6B. This is an eye-chart, and the distance to the chart was twenty-feet. In the original, line 7 of the chart could be easily read, indicated an effective "acuity" of 20/30, or about 1.5 minutes of arc.

The purpose of this work was to simulate a wide angle scene (about 100 degrees), roughly comparable to human vision, at human visual acuity. Figure 6C shows this scene, blurred by a space variant filter which is modeled after human visual acuity. Figure 6D shows the image of 6A, modeled in terms of a complex logarithmic model[7] of human visual cortex. The eye-chart occupies almost half of the surface of visual cortex, although it occupies a tiny fraction of the original scene. The ladder, and the windows of the original are compressed to almost the same size as the centrally fixated letters of the eye-chart. This illustrates the tremendous space variant compression of human vision. Variations in linear size of about $100^2{:}1$ (10^4 in solid angle) occur from the center to the periphery of the human visual system.

References

[1] B.M. Dow, A.Z. Snyder, R.G. Vautin, and R. Bauer, "Magnification factor and receptive field size in foveal striate cortex of monkey," *Exp. Brain Res.*, vol. 44, pp. 213-228, 1981.

[2] Martin Levine, *Vision in Man and Machine*, McGraw-Hill, New York, 1985.

[3] D. Noton and L. Stark, "Scanpaths in saccadic eye movements while viewing and recognizing patterns," *Vision Research*, vol. 11, pp. 929-942, 1971.

[4] Rayner, K., "Eye movement in reading and information processing," *Psychological Bulletin*, vol. 85, pp. 618-660, 1978.

[5] C.W. Richard and H. Hemami, "Identification of 3D objects using Fourier descriptors of the boundary curve," *IEEE Trans. on Systems, Man. and Cybernetics*, vol. SMC-4 No. 4, pp. 371-378, 1974.

[6] E.L. Schwartz, "On the mathematical structure of the retinotopic mapping of primate striate cortex," *Science*, vol. 227, p. 1066, 1985.

[7] E.L. Schwartz, "Computational anatomy and functional architecture of striate cortex: a spatial-mapping approach to perceptual coding," *Vision Research*, vol. 20, pp. 645-670, 1980.

[8] E.L. Schwartz, R. Desimone, T. Albright, and C.G. Gross, "Shape recognition and inferior temporal neurons," *Proceedings of the National Academy of Sciences*, 1983.

[9] R.W. Sperry, *J. Comp. Physiology*, vol. 43, pp. 482-489, 1950.

[10] R.B. Tootel, M.S. Silverman, E. Switkes, and R. deValois, "Deoxyglucose, retinotopic mapping and the complex log model in striate cortex." *Science*, vol. 227, p. 1066, 1985.

[11] D.C. VanEssen, W.T. Newsome, and J.H.R. Maunsell, "The visual representation in striate cortex of the macaque monkey: Assymetries, anisotropies, and individual variability.," *Vision Research*, vol. 24, pp. 429-448, 1984.

[12] E. Wolfson and E.L. Schwartz, "Space-variant image-processing II: A truncated-pyramid algorithm for _.ga-pixel image-warping," *Comp.Neuro.Tech.Rep.*, vol. CNS-TR-23-86, NYU Med. Ctr./Computational Neuroscience Laboratories, 1986.

[13] E. Wolfson, Y. Yeshurun, and E.L. Schwartz, "Space-variant image-processing II: Image-blending of multi-fixation logarithmic views," *Comp.Neuro.Tech.Rep.*, vol. CNS-TR-10-86, NYU Med. Ctr./Computational Neuroscience Laboratories, 1986.

Figure 1b

Figure 1a

191
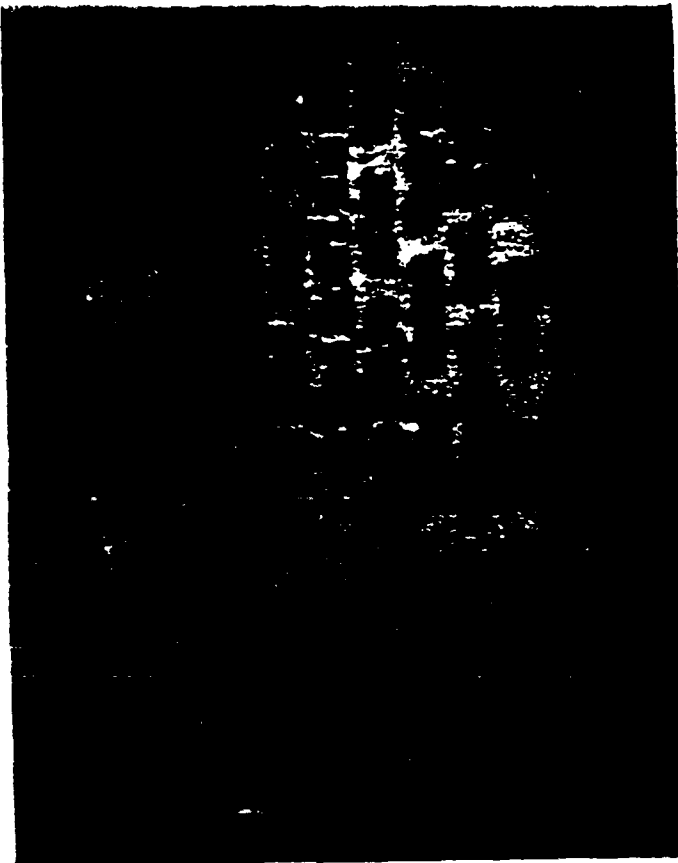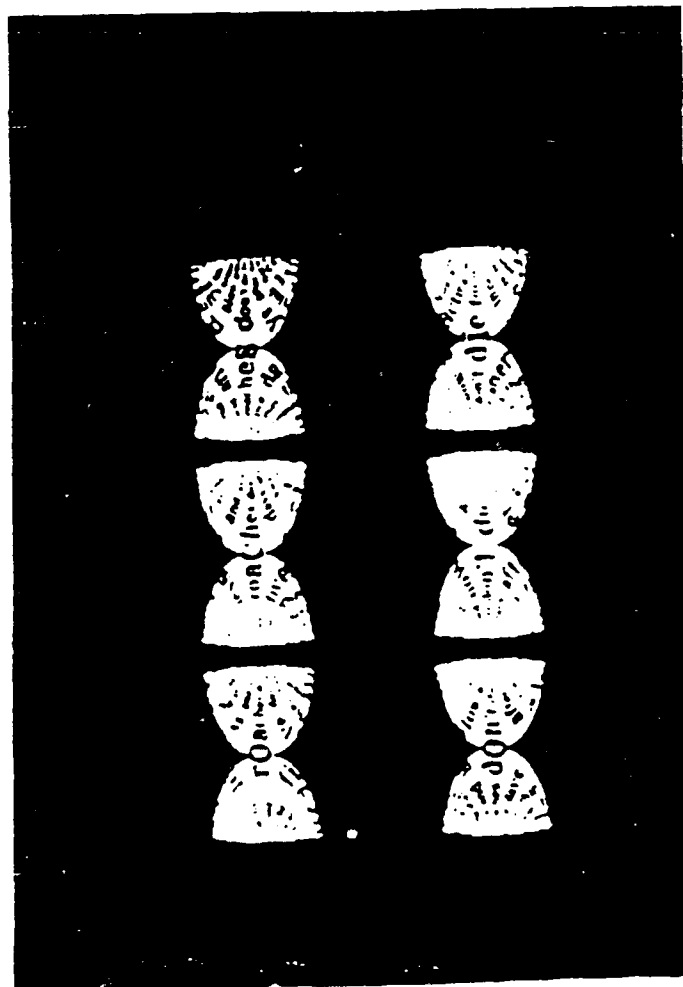
FIGURE 2b

a

b

c

d
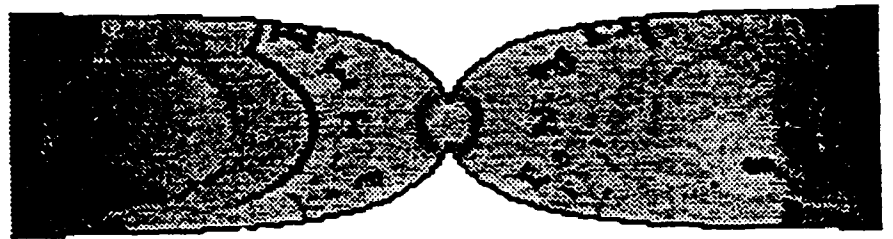
FIGURE 3b
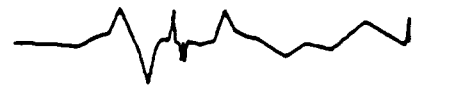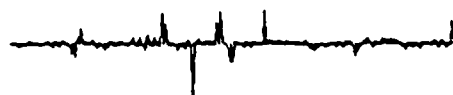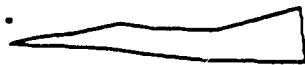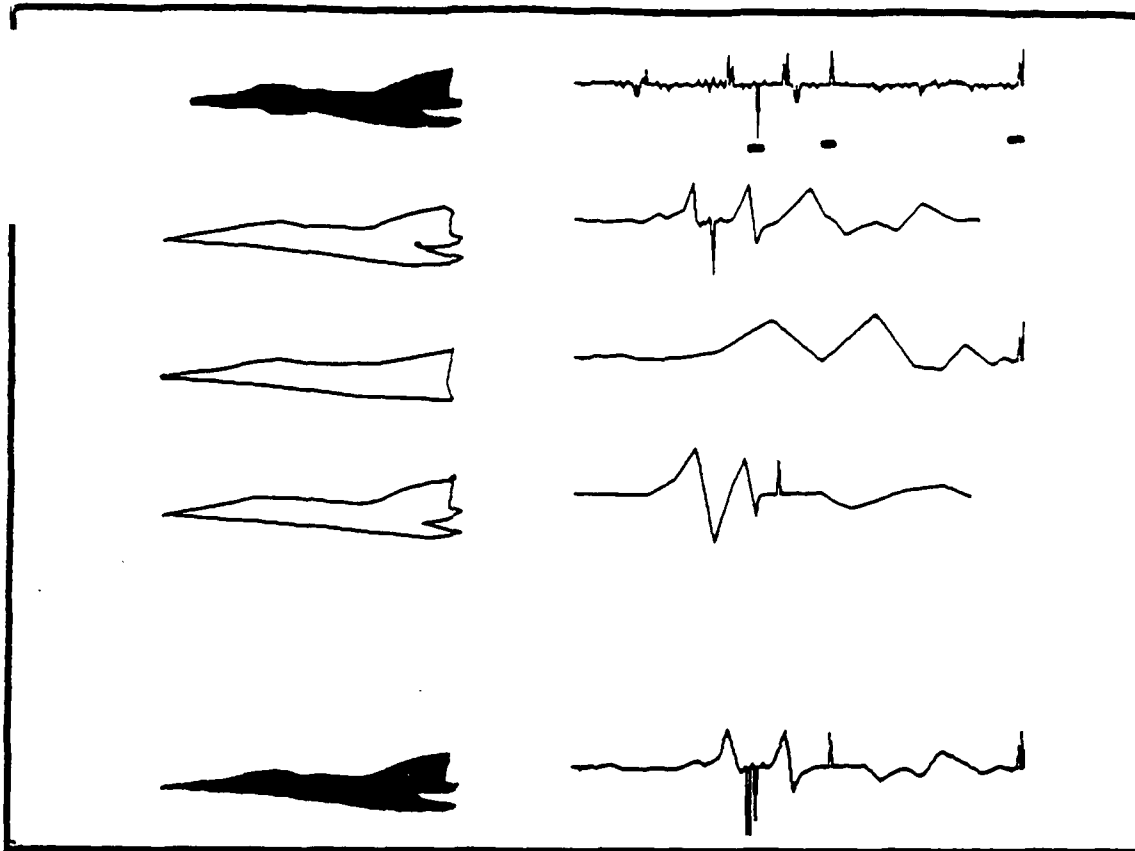
FIGURE 4b
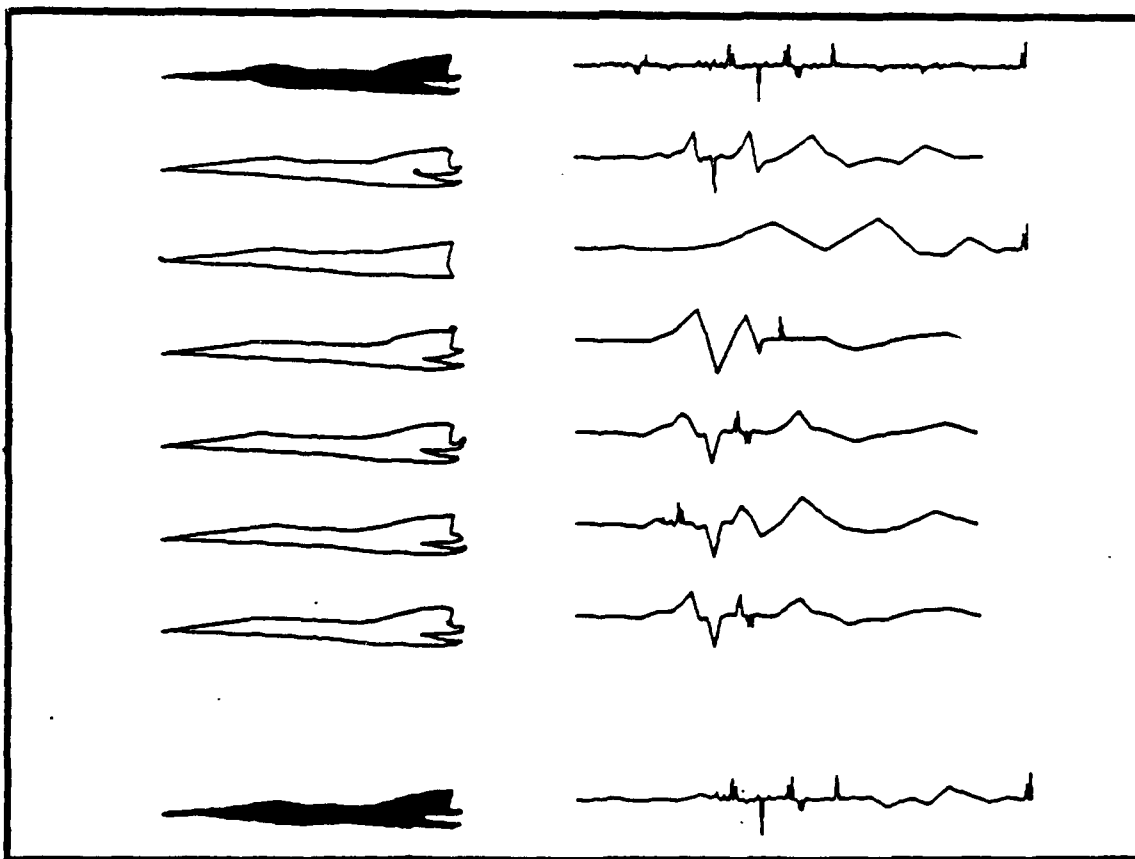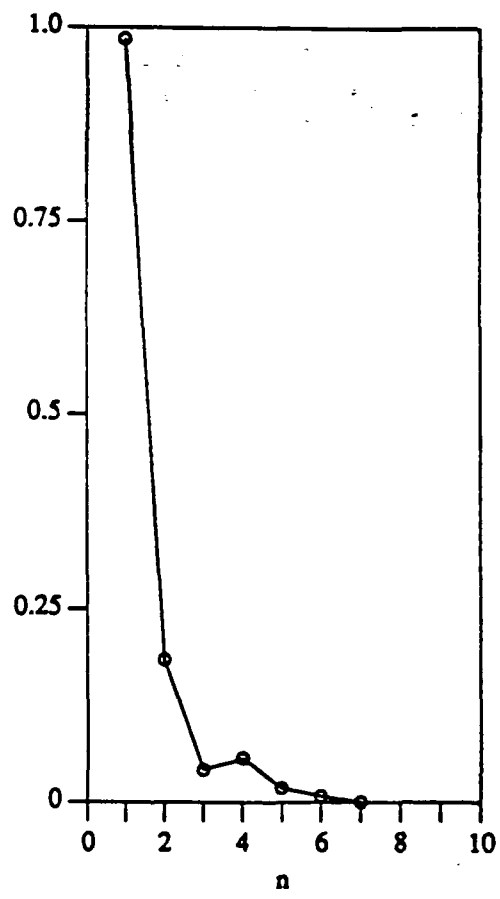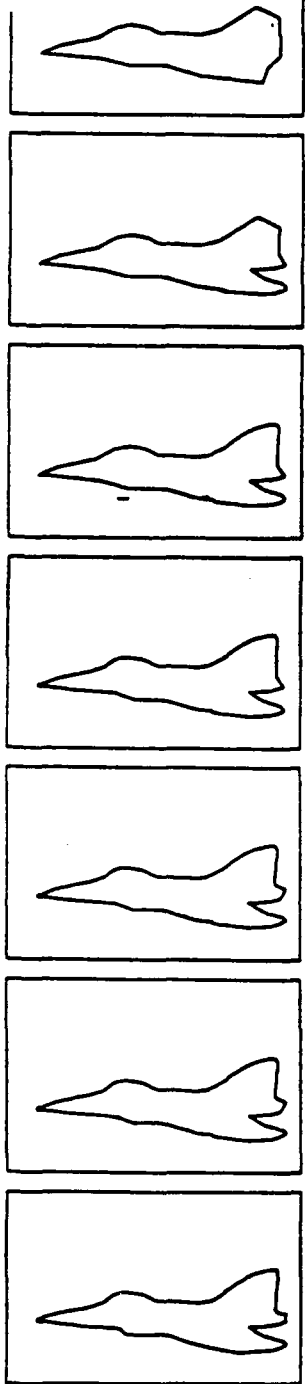
FIGURE 5

195

A
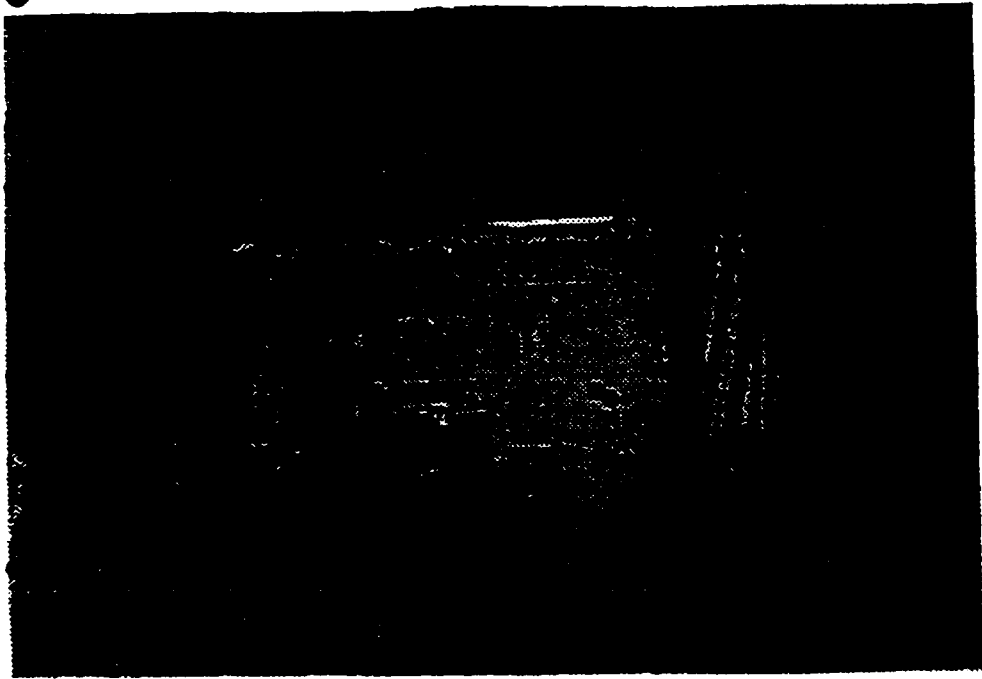
B
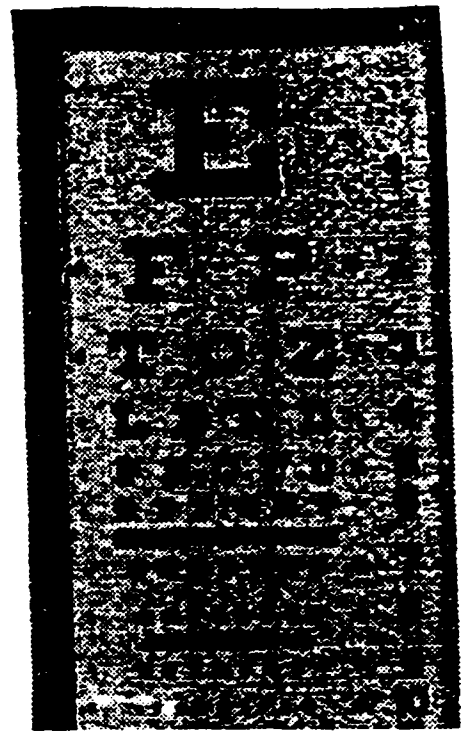
C

D

CHARACTERIZATION OF RIGHT-HANDED
AND LEFT-HANDED OBJECTS

Yaakov Hel-Or
Shmuel Peleg
Hagit Zabrodsky

Department of Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, Israel

Abstract

Many natural shapes have chirality (or handedness): for instance our hands have a right-hand version and a left-hand version, the two types being mirror images of each other. In chemistry, for example, molecules and crystals are classified as having chirality D or L. Interaction between molecules is dependent on their chirality, and chirality may determine chemical characteristics. For instance, only glucose of D-chirality is sweet, while glucose of L-chirality is tasteless.

We study the notion of chirality for two dimensional binary shapes, and introduce measures to test whether a shape is symmetric, and if not whether it is left-handed or right-handed. The measures are based on boundary analysis, and perform well even when digital images of left-handed shapes differ from the mirror images of right-handed shapes. Such situations may occur due to natural variations and digitization errors. The measures can also successfully treat partially occluded shapes, and provide indications on the change of chirality as resolution changes.

1. Introduction

Not only body parts have right or left handedness, this property, *chirality* , exist almost everywhere. Chirality has special significance in the study of elementary particles [1] whose chirality is due to their spin. Likewise molecules can appear in two possible configurations, called D (dextro) chirality and L (levo) chirality [2], each having different characteristics. For instance, glucose of D-chirality is sweet, whereas glucose of L-chirality is tasteless. The first to observe the importance of chirality in chemistry were the French chemists Louis Pasteur (1822-1895) and Jean Baptiste Biot (1774-1862) who determined the connection between crystal's chirality and the deflection of the plane of polarization light passing through them [3].

One property that characterizes chirality is that an object can not be superimposed on its mirror image using translation and rotation. A right hand will never be similar to a left hand unless we look at one of them through a mirror. Thus, the set of all human hands can be

---

divided into two classes, each having its own specific chirality.

The goal of our work is to examine a set of two dimensional shapes, and reveal whether the objects in the set are chiralic. Once shapes are found to be chiralic, we would like to classify them according to chirality class. Theoretically, it is enough to check whether an object has a reflective symmetry, as chirality is a form of asymmetry. However, almost no real object is exactly symmetric, especially after digitization, therefore we must determine weather the lack of symmetry is a dominant characteristic of the object.

Figure 1 exhibits some intuitive properties of this analysis. Shape A_1 is symmetric and non chiral since its mirror image, A_2, can be superimposed on it by using translation and rotation. Shape B_1, which is obtained by shortening one arm of A_1, is chiral. Shape C_1, with an even shorter arm, is also chiral to a greater degree than B_1. Shortening the arm completely to produce the straight line D_1 results in a symmetric shape again.

The rest of this section is devoted to some basic definitions. Sections 2 and 3 are a review of conventional approaches that seemed theoretically appealing for chirality analysis but were not successful. Section 4 describes our new approach to measure chirality.
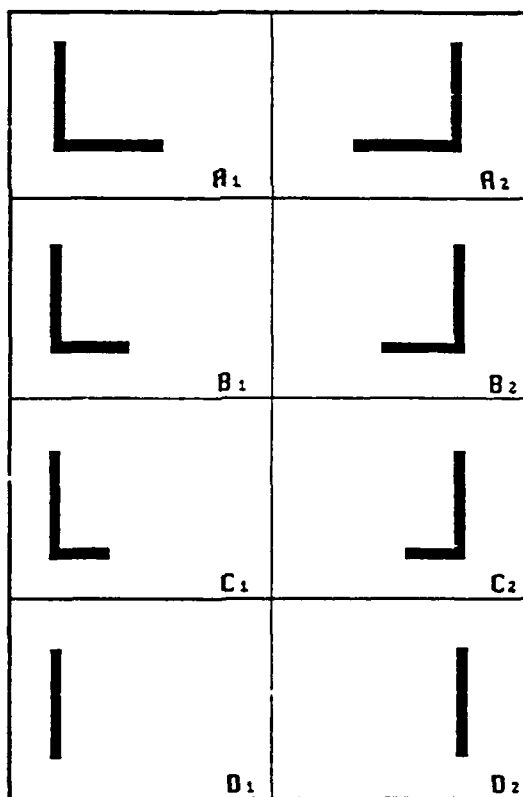

Figure 1:
Shape A_1 is symmetric, B_1 is chiral, C_1 even more chiral, and D_1 is symmetric.

1.1. Chirality

Let R be the set of points in the plane, and let $K \subseteq R$ be a set of points. K will be called *chiral* iff there are no reflection σ, translation τ, and rotation δ such that $\delta\tau\sigma(K) = K$. In other words: K is chiral iff it cannot be superimposed on its mirror image using only translation and rotation.

Let K be a chiral set, and let $\sigma(K) = K'$, i.e. K' is the mirror image of K. K and K' are called *enantiomers* and cannot be superimposed on each other.

1.2. Symmetry

K is *symmetric* iff there exist an isometry, which is not the identity, that transforms K onto itself (An isometry is a distance preserving transformation). Therefore, a set which is not chiral is symmetric.

K is *asymmetric* iff there is no isometry that transforms K onto itself.

K is *dissymmetric* iff there is no reflection that transforms K onto itself.

Note: A set is chiral iff it is asymmetric or at least dissymmetric. There are shapes, like the letter Z, that are symmetric, dissymmetric and chiral.

1.3. Centroid

Let $\chi: R \to \{0,1\}$ be the characteristic function of the set $K \subseteq R$,

$$\chi(x,y) = \begin{cases} 1 & \text{if } (x,y) \in K \\ 0 & otherwise \end{cases}$$

The *centroid* of K, (x_0, y_0), is the point such that

$$x_0 = \frac{\sum \chi(x,y)x}{\sum \chi(x,y)} \quad ; \quad y_0 = \frac{\sum \chi(x,y)y}{\sum \chi(x,y)}$$

where the summations above are over the entire plane.

It can be shown that a set K is not chiral iff there is a reflection σ that maps K onto itself. In this case the reflection is about a line that passes through the centroid of K.

2. Moments

The basic approach of using moments for shape analysis is developed by Hu [4]. Using the fact that a set is not chiral iff it is a reflection of itself about a line that passes through its centroid, we look for such a line. Since the centroid can be found easily, we only need to find the angle of this line, and then check the reflection about it.

Given the characteristic function $\chi(x,y)$, its M_{ij} moment is defined by

$$M_{ij} = \sum_{x,y} \chi(x,y) x^i y^j$$

We can find the centroid using $x_0 = \dfrac{M_{10}}{M_{00}}$, $y_0 = \dfrac{M_{01}}{M_{00}}$.

From now on we assume that the origin is in the centroid. If the axis of reflection coincides

with the y-axis then $M_{ij} = 0$ for odd i since $\chi(x,y) = \chi(-x,y)$. If the reflection axis coincides with the x-axis then $M_{ij} = 0$ for odd j. We will therefore rotate the shape about its origin until $M_{11} = 0$. In this case, if the set is symmetric, either the x-axis or the y-axis is the axis of reflection.

The effect of rotation by θ on M_{11}, yielding M'_{11}, can be shown to be

$$M'_{11} = \cos\theta \, (M_{02}\sin\theta + M_{11}\cos\theta) - \sin\theta(M_{20}\cos\theta + M_{11}\sin\theta)$$

Looking for θ such that $M'_{11} = 0$ we get

$$\tan(2\theta) = \frac{2M_{11}}{M_{20} - M_{02}} \tag{1}$$

The axis we get after moving the origin to the centroid, and then rotating by θ found in (1) is called the *principal—axis*. If the set is symmetric, it is now symmetric in respect to the x-axis or the y-axis, as $M'_{11} = 0$. If M'_{12} is very small then the y-axis is probably the reflection axis, and if M'_{21} is very small then the x-axis is probably the reflection axis (for exact symmetry either M'_{21} or M'_{12} equals zero). We can now measure the symmetry using correlation. If we assume that the y-axis is the axis of reflection, the measure is

$$W = \frac{\displaystyle\sum_x \sum_y (\chi^2(x,y) - \chi(x,y)\chi(-x,y))}{\displaystyle\sum_x \sum_y \chi^2(x,y)} \tag{2}$$

$W = 0$ indicates symmetric objects, and higher values (maximaly 1) indicate increased chirality.

Using expression (2) we can theoretically find chiral objects, but the results of this method on several shapes were found to be unreliable. Although theoretically the results should be accurate, in practice we used digitized images so that the results were not stable, and the method was found not to be robust. Furthermore, this analysis does not distinguish between enantiomers.

3. Transform Approach

Bigun and Granlaund [5] introduced a transform whose basis functions are spirals, with varying number of "arms" and curvature. Some of the basis functions are shown in Figure 2. As spirals are chiral, they can be used to measure chirality. Left spirals and right spirals have opposing chirality, while the border situation of "spirals" with straight hands is symmetric. Before describing the approach in detail we will mention that it is applicable to grey-level images as well as to binary images.

We will transform the shape $\chi(x,y)$ into polar representation,

$$\chi'(r,\theta) = \chi(r\cos\theta, r\sin\theta)$$

From now on we will represent our shape by a polar representation.
Let Ω be a filled circle of radius R, and let $f(r,\theta)$ and $g(r,\theta)$ be two functions on Ω. We define the scalar product of f and g, $<f,g>$, by

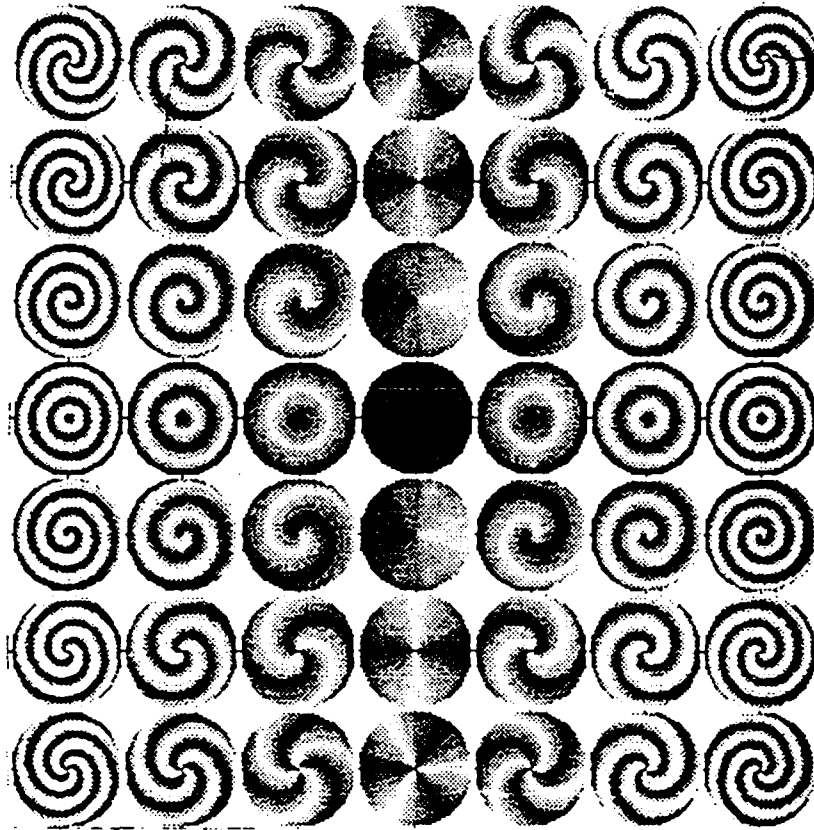$$<f,g> = \frac{1}{2\pi R} \int_0^{2\pi} \int_0^R f^*(r,\theta)g(r,\theta)d\theta dr$$

Figure 2:
Bigun's basis functions

We use the following set as the basis functions :

$$\Phi_{mn}(r,\theta) = e^{i(mwr + n\theta)} \tag{3}$$

where m,n are integers, and $w = \dfrac{2\pi}{R}$. This set is the one shown in Figure 2, and its arguments are as follows:

n - represents the number of arms.
m - represents the curvature of the arms.
$sgn(m \cdot n)$ - represents the direction of curvature (left spiral vs. right spiral). Due to this feature we need only consider $n > 0$.

The set (3) is a complete orthogonal set, and any continuous function on Ω can be represented by a weighted sum of its members. Φ_{mn} is actually the Fourier functions over the r,θ domain.

Let $f(r,\theta)$ be our shape function on Ω after we have transformed it into polar representation; then we can write

$$f(r,\theta) = \sum_{m,n} C_{mn}\Phi_{mn}(r,\theta)$$

where

$$C_{mn} = <f, \Phi_{mn}> = \frac{1}{2\pi R} \int_0^{2\pi} \int_0^R f(r,\theta) e^{i(mwr + n\theta)} d\theta dr$$

We use the coefficients C_{mn} to analyse an object's chirality after normalizing the image function such that for pure spirals, where $f(r,\theta) = a\Phi_{kl} + b$, then $C_{kl} = 1$ and all other C's are zero.

The following points should be noted:

- The results depend strongly on the choice of origin. Since we know that if an object is symmetric the symmetry axis passes through its centroid, we will use the centroid as the origin.
- The coefficients C_{mn} are complex. By using their magnitude, and neglecting the phase, the results are rotation invariant.

To find the chirality with respect to the origin we use the average of n and m weighted by C_{mn}:

$$M = \frac{\sum_m \sum_n |C_{mn}| m}{\sum_m \sum_n |C_{mn}|} \qquad N = \frac{\sum_m \sum_n |C_{mn}| n}{\sum_m \sum_n |C_{mn}|} \tag{4}$$

$abs(M)$ represents the magnitude of the chirality.
$sgn(M)$ represents the direction of the chirality.
N indicates the rotational symmetry as represented by the average number of arms.

This method was tried on a number of samples, but the results were unsatisfying. We found that noise disturbed the results. Further, the conversion into polar coordinates of a grid sampled image gave rise to inaccuracies.

4. Rotational Chirality Measures

Features based on object rotation can be used for chirality analysis. As clockwise rotation of an object is identical to counterclockwise rotation of its mirror image, non-chiral objects, which are identical to their mirror-image, will exhibit indifference to the direction of rotation. Chiral objects, on the other hand, will behave differently for the two directions of rotation.

In our scheme we use the following idea: imagine the object as rotating in a medium full of tiny particles. Some boundary segments will "collect" particles. We will use the length of these segments as a feature for chirality analysis. An ideal spiral, for example, rotated in one direction will have no "collecting" points, while rotation in the other direction will have all it's points "collecting". We will initially perform the rotation around the centroid, but eventually use other points. The choice of the center of rotation will be discussed later.

4.1. Boundary Based Measures

Let K be a set of points (pixels), and let E be the set of edge pixels of K, $E \subseteq K$. We will use subsets of the edge pixels that "collect" particles, RGP (right-grasp-pixels) and LGP (left-grasp-pixels), to define chirality measures. We assume that K is simply connected, and define the following:

Let $\{e_i\}_{i=1}^{k}$ be the sequence of boundary pixels ordered by following the boundary so that the object is to right [6], as in Figure 3. Let O be the axis of rotation. For a boundary pixel e_i we define:

- $\vec{r}_i$: the vector from O to e_i.
- d_i : the length of $\vec{r}_i$.
- θ_i : the angle between $\vec{r}_i$ and the x-axis.
- Δd_i : $d_{(i+1) \bmod k} - d_i$, the change in distance from O between e_i and e_{i+1}.
- $\Delta \theta_i$: $\theta_{(i+1) \bmod k} - \theta_i$, the change in θ between $\vec{r}_i$ and $\vec{r}_{i+1}$, the angle (e_i, O, e_{i+1}).

We represent the angles in the range $-\pi < \Delta\theta_i, \theta_i \leq \pi$. Figure 3 shows these definitions. Δd_i and $\Delta \theta_i$ can be positive, negative, or zero. When smoothing is desired, we can use $\Delta d_i = (d_{i+1} - d_{i-1})/2$ and $\Delta \theta_i = (\theta_{i+1} - \theta_{i-1})/2$.
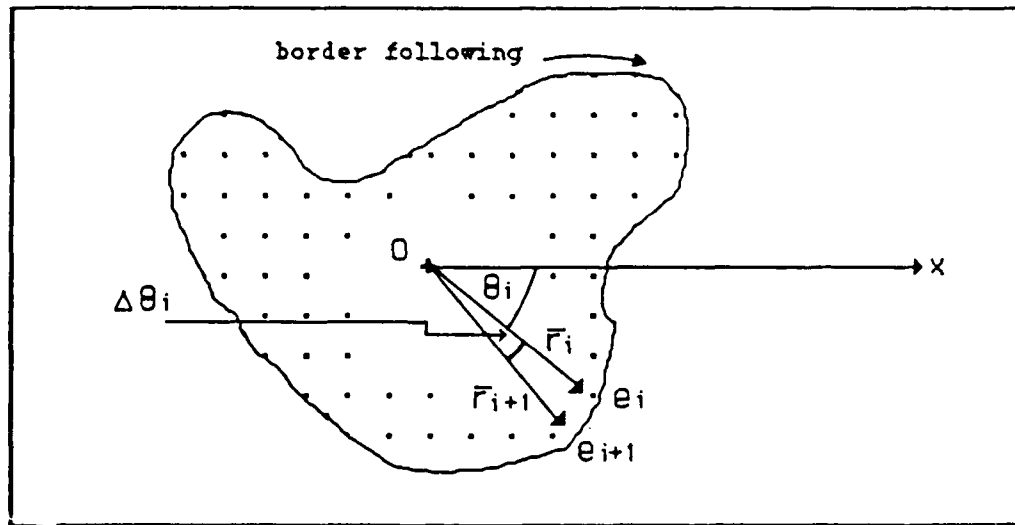

Figure 3:
some definitions on boundary pixels.

203

A boundary segment between e_i and e_{i+1} will be on the front edge, encountering particles in clockwise rotation, only if $\Delta d_i < 0$ (Figures 4.C 4.D) and in counter-clockwise rotation when $\Delta d_i > 0$ (Figures 4.A 4.B) The centrifugal power will push the particles away from the axis of rotation, unless the boundary itself serves as an obstacle when $\Delta \theta_i > 0$ (Figures 4.A 4.C). We therefore have

$$LGP = \{e_i \mid \Delta \theta_i > 0 \ , \ \Delta d_i > 0\} \tag{5}$$

$$RGP = \{e_i \mid \Delta \theta_i > 0 \ , \ \Delta d_i < 0\}$$

and we notice that $RGP \cap LGP = \varnothing$, and $LGP \cup RGP \subseteq E$. In practice we do not use only the signs of $\Delta \theta_i$ and Δd_i as in definition (5) since it can have very noisy behavior for small values. Therefore, for a given thresholds ε_1 and ε_2 we determine

$$LGP = \{e_i \mid \Delta \theta_i > \varepsilon_1 / d_i \ , \ \Delta d_i > \varepsilon_2\} \tag{6}$$

$$RGP = \{e_i \mid \Delta \theta_i > \varepsilon_1 / d_i \ , \ \Delta d_i < \varepsilon_2\}$$
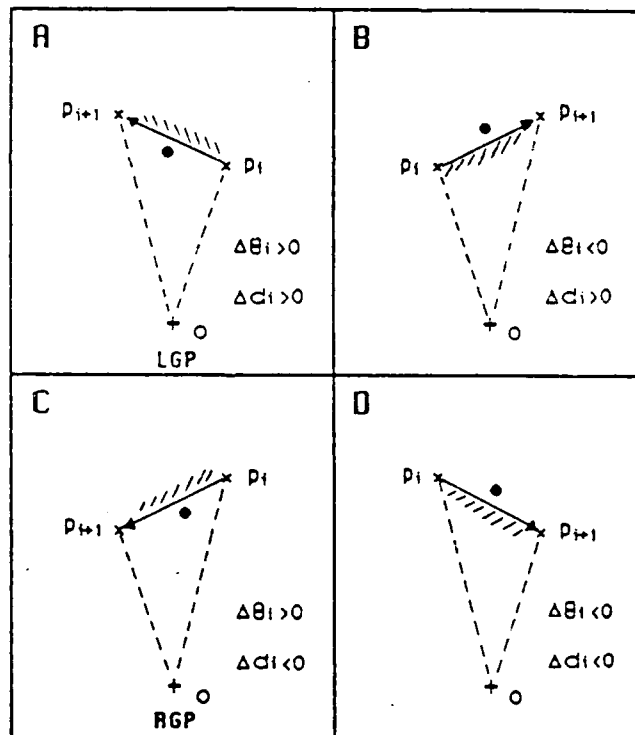

Figure 4:
properties of boundary particles in rotation.
4.A - Edge encountering and grasping particles in counter-clockwise rotation (LGP).
4.B - Edge encountering but pushing away particles in counter-clockwise rotation.
4.C - Edge encountering and grasping particles in clockwise rotation (RGP).
4.D - Edge encountering but pushing away particles in clockwise rotation.

As chirality measure we use the measure

$$Z = \frac{|LGP| - |RGP|}{|E|} \qquad (7)$$

where the normalization by $|E|$ serves to make the measure independent of size but dependent on the ratio of grasp-pixels to edge-pixels.

In order to develop another measure we adopt the idea of torque, which is force times the distance from the axis. Following this paradigm we can get a slightly different chirality measure: Let $L = \sum_{i \in LGP} d_i$, and $R = \sum_{i \in RGP} d_i$, then a chirality measure will be

$$Z' = \frac{L - R}{\sum_{i \in RGP,LGP} d_i} \qquad (8)$$

Figure 5 shows measures (7) and (8) applied to several shapes, when the centroid is used as the rotation axis. Notice that the shape in Figure 5.c is chiral, but since $|LGP| = |RGP|$ measure (7) fails to find its chirality, while measure (8) succeeds.
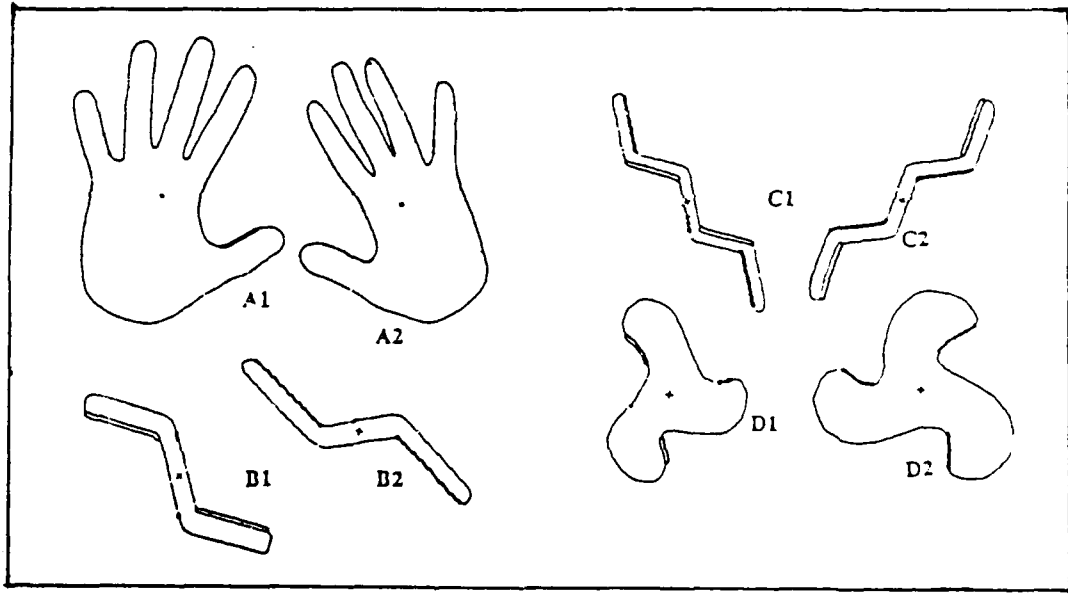

Figure 5:

Application of different rotational chirality measures on several shapes. The black squares $\in RGP$ and the white squares $\in LGP$.

picture	measure (7)	measure (8)
A1	0.02	0.88
A2	-0.02	-0.80
B1	0.26	0.94
B2	-0.27	-0.97
C1	-0.04	-0.25
C2	-0.01	0.23
D1	0.10	0.97
D2	-0.16	-1.00
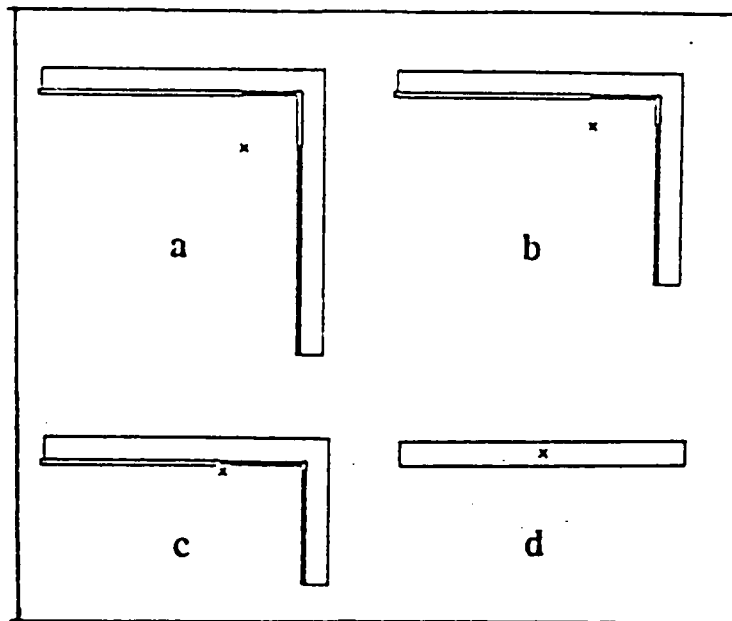
Figure 6:
Application of measure (7) to a series of shapes around the centroid.

picture	chirality-measure (7)
(a)	-0.001584
(b)	-0.005445
(c)	-0.006369
(d)	0.0

When we apply measure (7) to a series of shapes as in Figure 1 above, we obtain the predicted results which are shown in Figure 6. In Figure 6, (a) and (d) are not chiral, and indeed have minimum chirality measure. Examples (b) and (c) are both chiral, where (c) has more chirality than (b), and this effect too is reflected in the computed measurements.

4.2. Center of Chirality

Any chirality measure is greatly dependent on the choice of the axis of rotation. The centroid has initially been used as axis of rotation, but this choice can be misleading in some cases, especially for partially occluded shapes. Even for a spiral the centroid will not be the center of the spiral, as shown in Figure 7. We therefore define the following :
center of chirality is a point that maximize the rotational chirality measure ((7) in absolute value) when used as a rotation axis. Figure 7 shows the center of chirality for several shapes. It finds the correct center of the spiral, as well as the real center of some partially occluded shapes.

In order to reduce the computational complexity involved in the computation of the center of chirality, and avoid computing the chirality around every point of the image, several heuristics can be used. We could, for example, start searching for the maximal chirality at the centroid , examine a small neighborhood of the current location, and move to the pixel of highest chirality. This iterative search will stop when a point has higher chirality than all its neighbors. Simulated Annealing [7] can be used to prevent stopping at local maxima. A faster method to reach the center of chirality uses a multiresolution approach, and is discussed in the following section.
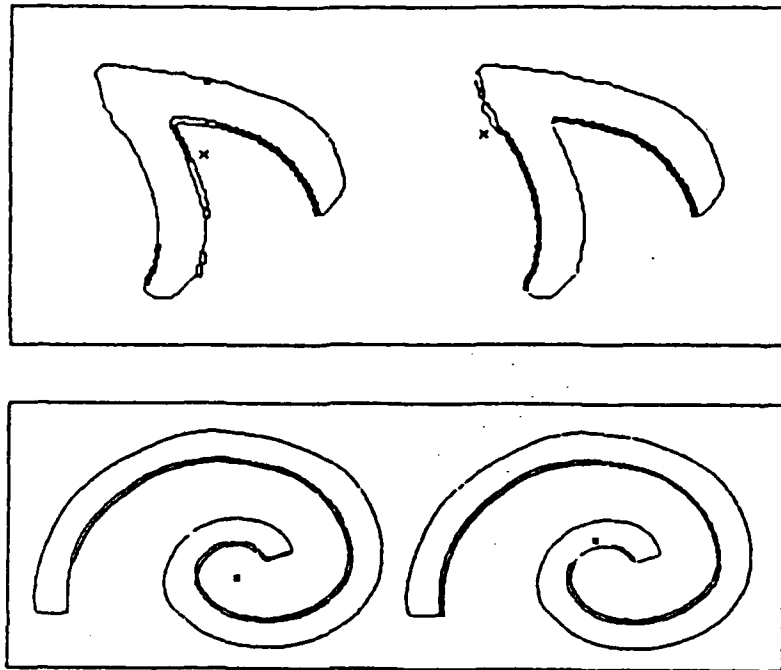
Figure 7:
The center of chirality (left) and the centroid (right) of some shapes

5. Multiresolution Approach

Define a pyramid [8] as a sequence of reduced resolution images. The lowest level of the pyramid, L_0, will be the original image of side length 2^N. L_1 will be a reduce image, having a side length of 2^{N-1}, etc. We use the pyramid multiresolution structure for speeding-up the computation and for measuring resolution-dependent chirality information.

The computation of the center of chirality in the pyramid is very fast. We start by computing the center of chirality at a high level using exhaustive search. This is very fast, as such level has only a small number of pixels. Let e_i be the center of chirality at level i. The center of chirality at level $i-1$ can now be computed by projecting e_i into level L_{i-1}, and searching for maximum chirality only in a small neighborhood around this projection. The speed-up introduced in this manner is of order $O(2^{2N})^2$, and uses the assumption that details added between levels L_i and L_{i-1} can change the location of the center of chirality only by a limited distance.

Computing the chirality measure at all resolution levels not only speeds up computation, but reveals information on the shape under consideration. The chirality at lower resolution levels describes a feature of the general shape, while chirality at higher resolution levels incorporates the features of the fine details. When the chirality of the fine details differs from the chirality of the general shape, the chirality measure can change drastically with resolution as shown in Figure 8.a . Figure 8.b shows another benefit of the multiresolution approach.

References

[1]   - M. B. Green: Superstrings, *Scientific American*, pp. 44-56, September 1986.

[2]   - H. H. Jaffe and M. Orchin : *symmetry in chemistry*, John Wiley and Sons, New York 1965.

[3]   - R. Dubos, *Pasteur and Modern Science*, Anchor 1960.

[4]   - M. Hu : Visual Pattern Recognition by Moment Invariants, *IRE Tran. on Information Theory*, pp. 179-187, 1962.

[5]   - J. Bigun and G. Granlund : Central Symmetry Modelling, Linkoping university, internal report of the department of electrical engineering, 1986.

[6]   - A. Rosenfeld and A. C. Kak : *Digital Picture Processing*, vol. 2, Academic Press, New York, 1982.

[7]   - S. Kirpatrick and C. D. Gelatt, Jr. , M. P. Vecchi : Optimization Simulated Annealing, *science*, pp. 671-680, 13 May 1983.

[8]   - A. Rosenfeld : *Multiresolution Image Processing and Analysis*, Springer-Verlag, 1984.
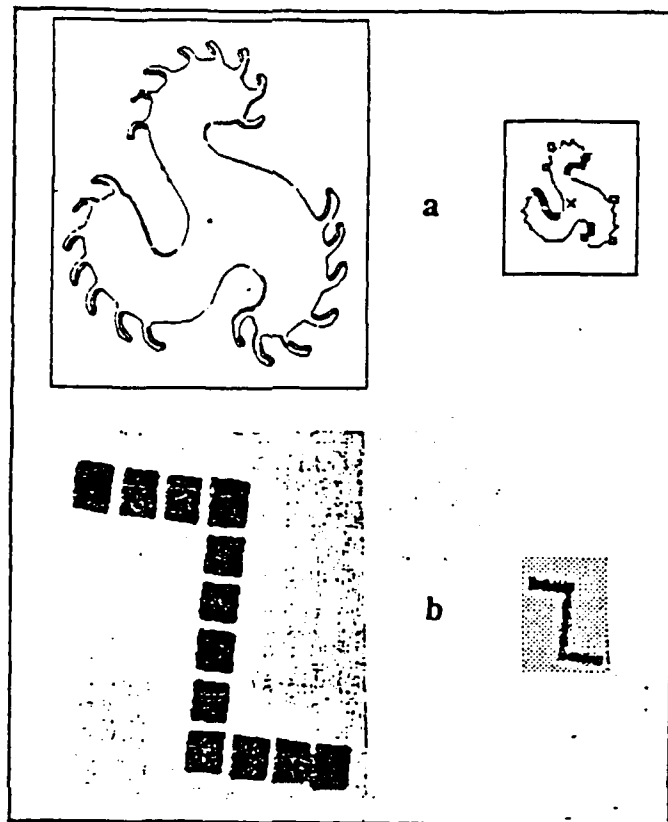
Figure 8:
Multiresolution Chirality Analysis.
a) Different chirality for general shape at low resolution and details at high resolution.
b) Disconnected object that becomes connected at lower resolution level.

The pyramid can also help in the analysis of non connected objects. The rotational measures give desired results only on simply connected objects. When fragmented objects are given, the reduction of resolution can yield connected objects at lower resolution level, where analysis is possible. Figure 8.b shows the analysis of non-connected object at lower resolution.

6. Concluding Remarks

A measure based on rotational features of two dimensional objects has been suggested for chirality analysis. This measure is robust, and is immune to insignificant deviation and some occlusion. It has a drawback in that it works only for simply connected binary shapes, as compared to the transform and moments methods, which are theoretically applicable to every function. However, in its domain it has superior performance than the other methods.

Acknowledgment

USING SIMPLE FEATURES FOR 3-D OBJECT RECOGNITION

Jezekiel Ben-Arie
Dept. of Aeronautical Engineering, Technion - Israel Inst. of Technology
Haifa 32000, Israel

Abstract

We are discusing in this paper the use of simple primitives such as specific points, curved and straight edges, planar surfaces, angles and distances to 3-D object recognition from monocular images. The paper is divided into two sections. In the first section we describe a general recognition method which is based on optimal matching of multinary graphs by ordered search algorithm. An admissible structure of the search guiding cost function combines error criteria based on hypothesized multinary geometrical relations with labeling probabilities that are obtained from other information sources. The geometrical cost is an error criterion which reflects through the disparity of the observation parameters, or by the mismatch of the projected model features, the consistency of partial matching of image-model feature subsets. The disparities were estimated using three kinds of features and methods: the area ratio method with planar surfaces, the directional method with linear features and the projection matrix with the specific points features.

In the second section we describe two novel probabilistic models of viewed angles and distances. These are derived using the "Observability Sphere" method. We conclude from these models that there are high prior probabilities that projected angles and distances have similar values to their 3-D sources. We employ these models for the recognition of 3-D objects by stochastic labeling.

A. Recognition of 3-D Objects by Optimal Matching of Multinary relations Graphs

A.1 Introduction

We are addressing here the problem of 3-D object recognition from 2-D monocular images. For the recognition process it is required to match projections of a stored library of 3-D models against a given monocular image. Actually, the problem is to determine if any of the library models could produce a portion of the image. The viewed object can have arbitrary 3-D position, scale and orientation and may be partially occluded. In order to recognize objects one has to match each of the models to the image and to find the best match with respect to some quantitative similarity criterion. The matching process has to overcome many obstacles: the lack of 3-D information in monocular images, the unknown position, scale and orientation of the object and its partial occultation.

One may overcome some of these obstacles by equivalent graph representation of the 3-D object model and the 2-D image and employ relational graph matchings in the recognition algorithm. In the recognition process, it is also necessary to rely on primitive elements and relations (i.e. graph nodes and arcs) which maintain their general properties under imaging transformations. Such an approach can be found in studies [1,4] where segmentation of objects into primitives such as generalized cylinders, sticks, plates and blobs, was carried out.

The above approach has some limitations:

(i)   3-D primitives hardly exist as whole entities in the image due to partial occultation, nonuniform albedo or illumination, etc. In addition, segmentation results of true images show that a sizeable portion of this information is missed.

(ii)   3-D primitives maintain their general properties only within a limited range of viewing orientations.

(iii)   Only simple relations (usually binary) are invariant under projection transformations. Whereas multinary non-invariant relations such as geometric ones are important, when objects of similar structure have to be differentiated.

Our general approach [7-11] which is also expressed in the present study, is different from the distortion invariant graphs approach mentioned above in the following aspects:

(i)   The projection originated deformations are utilized in the recognition process and not eliminated. These deformations are referred to as geometric error or geometric cost, which serves as a part of the search guiding function to the matching algorithm. The error will be low if the spatial relations of the image conform to the projected relations of the model and high otherwise.

(ii)   The method is based on low dimensionality primitives such as specific points and edges which are projection invariant and preferred for practical image segmentation.

(iii)   The method combines many kinds of information sources expressed in binary and multinary relations.

Generally, an objective evaluation with respect to any criterion is attained only if the matching algorithm yields optimal results. Thus, optimal matching has an important advantage over algorithm dependent methods. Another benefit of the optimal approach is that suboptimal but less complex algorithms may be derived [11] and properly assessed by controlled degradations of the optimal method. A number of graph optimal monomorphism algorithms have been suggested in the literature [2,3]. However, they are not applicable to multinary graphs matching.

Our matching algorithm which ultimately finds the optimal multinary graph matching is based on ordered search A* with few modifications. The matching procedure is as follows. First, the image and the object are segmented to relatively simple primitive elements: specific points, straight edges and flat regions which are preferred for practical image segmentation. Then, all the hypothesized partial matches of subsets of pairs of image/object primitives are implicitly organized as states in a combinatorial state space. A cost function is associated with each state. This cost function

combines the geometric cost with optional non-geometric information extracted from the image/model by relaxation or other methods. It should be noted that the geometric cost reflects the maximal disparity of the state's observation parameters (in the case of the projection matrix method it reflects the mismatch of the projected model to the image features).

Next, homomorphic (or monomorphic) [7] trees are constructed in the state space within which the cost guided ordered-search procedure is carried out. The cost function is constructed in a min-max fashion to ensure the admissibility conditions of the ordered search within the trees. By these steps the optimal (minimum cost) multinary graph matching of object/image graphs and a quantitative similarity of the model to image is found.

Each of the three kinds of primitive elements mentioned above can be employed in the recognition process by using a different geometric disparity criterion. For the flat regions primitives we developed an "area-ratio based method" which enables us to compute the viewing parameters and their geometric disparity. Another method the "directional method" was developed for the straight edges using only their orientations information. For the specific points we introduced a disparity criterion which is tolerant to scale alterations of each of the object axes independently. Thus, this criterion enables us to recognize generic objects in the sense of scale.

The search algorithm, implemented on complex objects, demonstrates a significant reduction in the average complexity (usually exponential) of the graph matching.

A.2 General Method and Cost Function Definitions

The matching procedure is performed between a prototype object which is a member of a library of stored models, and a candidate image. The prototype object set consists of L primitive parts denoted as labels $\{\lambda_k\}$. Each label λ_k is a node of the stored model graph. In a similar manner, the candidate object in the image consists of N primitive parts called units denoted by $\{u_i\}$. The units are the nodes of the image graph.

Let the units set U be defined as $U = \{u_i ; i = 1,...,N\}$ and the labels set A defined as $A \triangleq \{\lambda_k ; \quad k = 1, ...L ; \quad \cup \lambda_\phi\}$ where λ_ϕ is the empty label.

We define the "matching space" D, a state space in which each state n is defined uniquely by a subset of μ pairs called homomorphic match $F_\mu(n)$:

$$F_\mu(n) = [(u_{i_1}, \lambda_{k_1}), \quad (u_{i_2}, \lambda_{k_2}) ... (u_{i_\mu}, \lambda_{k_\mu})] \subset (UxA)^\mu$$

(1)

$$u_{i_a} \neq u_{i_b} \quad \text{if} \quad a \neq b ; \quad u_i \neq u_\phi$$

$F_\mu(n)$ can be regarded as a μ-nary relation where μ is the degree of the match. The matching space D includes all the possible combinatorial partial matches of unit-label pairs. The empty state n=0 which is defined by: $F_0(0) = (u_\phi, \lambda_\phi)$, where u_ϕ is the empty unit, is also included in D i.e.:

$$D \overset{\Delta}{=} F_\mu(n) \cup F_0(0) \subset \{(u_\phi, \lambda_\phi), (U x A)^1, \ldots (U x A)^N\} \qquad (2)$$

A cost function C(n) is defined over D. C(n) is a function only of the state n. The cost of the empty state will be defined as $C(0) \overset{\Delta}{=} 0$. Let n_t be the state of the best match in D. If all the units have to be matched, the degree of the match $F_\mu(n_t)$ is N: $F_\mu(n_t) = F_N(n_t)$ and the cost $C(n_t)$ is the lowest among all the states with degree N in D.

The geometric cost of $C_g(n)$ of a match $F_\mu(n)$ is defined in the following manner:

$$C_g(n) = \sum_{\rho=\rho_0}^{\rho_1} \beta_\rho \max_{F_\rho(n) \varepsilon F_\mu(n)} \sum_1^{m_a} C[F_\rho(n)] \qquad (3)$$

Instead of computing the total geometrical cost of $F_\mu(n)$, the match $F_\mu(n)$ is divided into partial ρ-nary relations $F_\rho(n)$ and for each ρ only the m_a highest disparities $C[F_\rho(n)]$ are taken into account. The β_ρ are the summing weights given to these costs. The constants ρ_0 and ρ_1 and m_a were not limited, the cost would be in favor of low degree matches, and the ordered search C* would degenerate to breath first search. The number of disparity terms in (3) is constant at tree depths greater than ρ_1, thus enabling efficient search. Another reason for this particular form of $C_g(n)$ is due to the admissibility condition of the search C* algorithm that is explained in [7].

The total cost function C(n) is constructed as the weighted sum of two components: $C_e(n)$ the labeling error probabilities cost and $C_g(n)$ the geometrical cost:

$$C_e(n) = \alpha C_e(n) + (1 - \alpha)C_g(n) \qquad 0 \le \alpha \le 1 \qquad (4)$$

The reason behind this mixture of error probabilities and geometrical errors is that both are monotonic functions of the matching quality. This fact allows us to incorporate geometric and non-geometric information sources that exist in the image. The factor α in (4) determines their relative importance.

The error cost $C_e(n)$ of an error in tha labeling assignments of $F_\mu(n)$, is the result of other labeling algorithms which combines, in this case, non-geometric featural and contextual data into a set of vector probabilities P_i. Various labeling algorithms may be used for the estimation of P_i. These can be grouping methods such as curve matching [6,11] or stochastic labeling. The stochastic labeling algorithm [13] is based on a relaxation procedure which finally attaches to each unit u_i a probability vector $P_i = [p_i(\lambda_1) \ldots p_i(\lambda_L)]^T$. Each of the vector components $P_i(\lambda_k)$ is the probability that the unit u_i matches the label λ_k.

The error cost $C_e(n)$ was constructed in the form:

$$C_e(n) = 1 - \min_{(u_i, \lambda_{k_i}) \in F_\mu(n)} [P_i(\lambda_{k_i})] \qquad (5)$$

that is admissible in the search [7].

The ordered search C* is conducted on homomorphic or monomorphic trees [7] that are defined within the space D. It is shown [7] that the homomorphic tree B contains the optimal node n_t in D, and that the cost function min-max structure always satisfies the admissibility conditions within the trees. Thus, the search is guaranteed to find the optimal state n_t in D.

For the first ρ_0-1 depth stages of the tree B the geometric cost is null because it requires at least ρ_0 pairs in the match $F_\mu(n)$. Usually ρ_0 and ρ_1 which are of the order of 4-6 are quite small compared to N. Therefore, it is suggested that the search C* will be divided into two stages. At the first stage, the search is based on the probabilistic information alone. The weight factor a is set to $\alpha=1$ and the search is conducted only on the initial ρ_0-1 stages of B tree. The results of the first stage a set D_0 of few best matches $D_0 = \{F_{\rho_0 - 1}(n_j)\}$ which are to be considered as initial predecessors for the second search stage. At the second search stage the full cost (4) is used until the optimal match is found.

A.3 Central Projection Geometric Criteria

In this section we describe a method [10] which enables the computation of geometric disparity measure. The physical meaning of the geometrical disparity in this section is the sum of squared distances between the image's specific points features to the projected models points. The computation of the disparity and the projection parameters here is simple and simultaneous.

Let a subset $\{\alpha_i; i = 1,...,\mu\}$ of model specific points be hypothetically matched to image specific points subset $\{\beta_j; j = 1,...,\mu\}$. Following the notation of [11] $\alpha_i = [x^i, y^i, z^i, 1]^T$ is expressed in homogeneous coordinates $\beta_j = [u^j, v^j, 1]^T$ the central projection relation can be expressed as

$$TA = B - E ; \quad A = [\underline{\alpha}_1, ..., \underline{\alpha}_\mu] ; \qquad B = [\underline{\beta}_1, ..., \underline{\beta}_\mu] \qquad (6)$$

E is the error matrix, T is the projection matrix and is a product of rotation, translation and scale matrices of the form :

214

$$
T = \begin{bmatrix} d_1 a_1 & d_2 a_2 & d_3 a_3 & x_0 \\ \\ d_1 a_4 & d_2 a_4 & d_3 a_4 & z_0 \\ \\ \dfrac{d_1 a_7}{f} & \dfrac{d_2 a_8}{f} & \dfrac{d_3 a_9}{f} & \dfrac{y_0 + f}{f} \end{bmatrix} \tag{7}
$$

where a_i ; $i = 1,...,9$; are Euler matrix coefficients, x_0, y_0, z_0 are camera translations, d_i ; $i = 1,2,3$; are the axes scale factors and f is the camera focal length. Minimal error norm is obtained when

$$
T = B A^T (A A^T)^{-1} \tag{8}
$$

The geometric disparity $C_g(n)$ which is:

$$
C[F_\mu(n)] = tr(E E^T) = tr[B(I - A^T(A A^T)^{-1}A)^2 B^T] \tag{9}
$$

reflects the quality of the μ-nary match $F_\mu(n)$. $C[F_\mu(n)]$ of this form is tolerant to scale changes.

This approach is similar to the alignment approach [4] in which the model is back projected to the image plane after initial estimation of the imaging parameters. Here we are computing the geometric disparity $C(F_\mu(n)]$ which is proportional to the sum of squared distances of the back projected model points to their respective image points. The expression of (9) was used for fast similarity measure in references [10,11] for a banch and bound C* search algorithm. The search results are given in table 1.

A.4 Experimental Results obtained by the projection matrix method

In the present study the search algorithm with the combined geometric-probabilistic cost function is tested here for recognition capability and for differentiation power. The test is performed on 4 aeroplane models (a SAAB, a KFIR, a F15 and a MIG25 - see Figs. 2A-2D respectively) and a Hercules C-130 aeroplane (Fig. 1). All the objects were segmented for specific points primitives from their true images by means of preprocessing procedure that is described in detail in [11].

The results of the search are summarized in Table 1. Each of the planes models and images are matched one against the other to test the recognition and differentiation power of the matching method. The rows of Table 1 relate to the images of the 4 planes and the columns to their models. In Table 1 one can also find the number of node examinations N_e required to reach the optimal match, number of matching errors M_e, (i.e., number of wrong labelings, such as assigning a wing tip of a model to a nose in the image), the final cost function $C(n_t)$ and the CPU time required. The notation "exp" which appears in some of the cases instead of a numerical values means that the cost rose to values that indicated that the search will terminate only at very high values of N_e. Such cases occur when one tries to match totally incompatible model/image pairs. Though, the time (or node examinations) required to detect incompatible matching is relatively short due to the steep rise in the cost function. From Table 1 it is evident that the similarity of aeroplane pairs such as F15-MIG25 and KFIR-SAAB is detected.

Table 1 : Cross Matching Results of 4 Aeroplanes.

Images \ models	SAAB	KFIR	F15	MIG25
SAAB	$N_e = 97$ $M_e = 0$ $C(n_t) = 12.34$ CPU = 0:1:02	$N_e = 373$ $M_e = 1$ $C(n_t) = 66.27$ CPU = 0:04:30	exp	exp
KFIR	$N_e = 973$ $M_e = 1$ $C(n_t) = 65.5$ CPU = 0:20:43	$N_e = 193$ $M_e = 0$ $C(n_t) = 13.10$ CPU = 0:01:46	exp	exp
F15	exp	exp	$N_e = 341$ $M_e=0$ $C(n_t) = 15.74$ CPU=0:17:11	$N_e=4178$ $M_e = 6$ $C(n_t)=71.7$ CPU=0:33:15
MIG25	exp	exp	$N_e = 613$ $M_e=2$ $C(n_t) =21.78$ CPU=0:51:39	$N_e=341$ $M_e = 1$ $C(n_t)=14.7$ CPU=0:13:50

The tolerance of the geometric cost in (3) to scale changes was tested with the Hercules model. The model was matched to two images (see Figs. 1A, 1B). The first is its regular image. In the second image the longitudinal axis scale was reduced to 50% of the original one. Except for the $C(n_t)$ due to the scale change, the results of N_e, M_e, and CPU were the same for the two cases: $N_{e1} = N_{e2} = 225$, $M_{e1} = M_{e2} = 0$, $CPU_1 = CPU_2 = 7.40$ mins.

These results demonstrate the generic object recognition capability of our optimal recognition method.

In the next two sections we introduce methods of imaging parameters estimation that are based on edges and regions. These primitives consist of large groups of picture elements and therefore their attributes are statistically less susceptible to noise.

A.5 Imaging Euler Angles Computation by Edge Orientations Based on Central Projection

In many practical cases, after segmentation has been carried out, straight edges appear in the image without their authentic endpoints (e.g. aeroplane wings with curved tips). In such cases the endpoints can not be considered as "specific points". On the other hand, the information regarding the direction is based on a large number of points and therefore is reliable and should be employed.

The spatial orientation of an object-edge is denoted by the unit vector a_i (see Fig. 1c). a_i is given in the object's coordinate system (x', y', z'). The transformation M (Euler Matrix) to the image coordinate system (x,y,z) should be sought. The central projection of a_i to the image plane is denoted by b_i. The normal to the plane which contains b_i and the center of projection c, is denoted by t_i. Both t_i and b_i can be specified in the (x,y,z) system. M is computed here by the following equation system:

$$t_i^T M a_i = 0 \qquad\qquad i = 1, 2, 3, \qquad\qquad (10)$$

The Euler angles vector $\Omega = [\psi, \theta, \phi]^T$ components which appear as variables of trigonometric functions in M, are non-linearly related and therefore must be evaluated by iterative computation as described below.

First, we express the full differential of M in the neighborhood of the initial angular position $\Omega_0 = [\psi_0, \theta_0, \phi_0]^T$. Then, one obtains for i = 1,...,n:

$$t_i^T M a_i = t_i^T M_0 a_i + t_i^T \left[\frac{\partial M}{\partial \psi} \bigg|_{\underline{\Omega}_0} a_i d\psi + \frac{\partial M}{\partial \theta} \bigg|_{\underline{\Omega}_0} a_i d\theta + \right.$$

$$\left. + \frac{\partial M}{\partial \phi} \bigg|_{\underline{\Omega}_0} a_i d\phi \right] = D_i + t_i^T dM_i d\underline{\Omega} \qquad i = 1, 2, \ldots, n \qquad (11)$$

where D_i is the following scalar.

$$D_i = t_i^T M_0 a_i \qquad (12)$$

and the matrix dM_i is obtained by reorganization of the vector columns as follows:

$$dM_i(\underline{\Omega}) = \left[\frac{\partial M}{\partial \psi} \bigg|_{\underline{\Omega}} a_i \quad \frac{\partial M}{\partial \theta} \bigg|_{\underline{\Omega}} a_i \quad \frac{\partial M}{\partial \phi} \bigg|_{\underline{\Omega}} a_i \right] \qquad (13)$$

Next $d\underline{\Omega}$ is computed via:

$$A d\underline{\Omega} = \begin{bmatrix} t_1^T dM_1 \\ \vdots \\ t_n^T dM_n \end{bmatrix} d\underline{\Omega} = \begin{bmatrix} -t_1^T M_0 a_1 \\ \vdots \\ -t_n^T M_0 a_n \end{bmatrix} = \begin{bmatrix} -D_1 \\ \vdots \\ -D_n \end{bmatrix} \stackrel{\Delta}{=} -D \qquad (14)$$

The optimal angular increment $d\hat{\underline{\Omega}}$ (in mean square error sense) is expressed by :

$$d\hat{\underline{\Omega}} = -(A^T A)^{-1} A^T D \qquad (15)$$

As to the iterative process, one starts with an arbitrary initial value $\underline{\Omega}_0$ which is updated according to $\underline{\Omega}_0 = \underline{\Omega}_0 + d\hat{\underline{\Omega}}$, until satisfactory deviation $\| D \|$ is reached. Convergence is obtained within 4-12 iterations.

A.6   Observation vector computation by area ratios

In this section we describe a method which enables the computation of the observation vector x from triples of area ratios of hypothetically matched image regions to planar faces of the object. These "faces" do not have to be real surfaces of the object and can also be any imaginary triangulation plane connecting three specific points of the object.

Let z_1, z_2, z_3 be 3 areas of planar faces of the object with their unit normals d_1, d_2, d_3, respectively. If these faces are matched to 3 image regions with areas y_1, y_2, y_3 respectively, a set of 3 equations is formed assuming the orthographic projection approximation:

$$s^2 z_i (d_i.x) = y_i \qquad\qquad i = 1,2,3,; \qquad\qquad (16)$$

where s is the projection's scale factor.

If d_1, d_2, d_3 are linearly independent than x has a unique solution based on the following equation set:

$$(d_i.x) / (d_j.x) = y_i\, z_j / y_j\, z_i \qquad i,j = 1,2,3 ; \quad i{\neq}j \qquad\qquad (17)$$

As shown here, this method results in a linear equations set and is economic computation wise. Other methods for the computation of imaging parameters based on orthographic and central projections appear in [11].

A.7 The Cost Function and Search Algorithm

In this section the general structure of cost function used in the search procedure is given. A detailed description of the subject has been presented recently by the authors in [11]. Here, only a brief discussion will be given. Thus, many of the details given in [11] will be avoided here.

The geometrical cost $C_g(n)$ of a state $F_\mu(n)$ of D must reflect the mismatch, or the error, that the partial matchings contained in $F_\mu(n)$ create by projecting the models primitive set onto the image plane. For instance, if the state n has a degree of four: $F_\mu(n) = F_4(n)$, it contains 4 pairs of matchings. If we are dealing with planar faces as labels of the model , and image regions as units, each three pairs will define an observation vector by (17). Thus, 4 pairs define 4 observation vectors, these are aligned if the match is exact, and misaligned if the pairs do not match or the data is degraded.

We define a parameter's vector w, which reflects the observation parameters used in the matching procedure. If the directional method in section A.5 is employed, then $w = \Omega$, and if the area ratio method of section A.6 is preferred, then $w = x$. Thus, one way to define a geometric disparity function $C[F_\mu(n)]$ is to measure the disparity of the observation parameters vectors w set $W_\mu(n)$ defined by all the triple pairs included in the μ-nary match $F_\mu(n)$:

$$C[F_\mu(n)] = \frac{1}{6} \sum_{(w_i, w_j) \epsilon W_\mu(n)} [1 - (w_i \cdot w_j)] \qquad\qquad (18)$$

where:

$$\gamma = \frac{\mu!}{(\mu - 3)! \, 3!} \; ; \quad \delta = \frac{\gamma!}{(\gamma - 2)! \, 2!}$$

An alternative method for the estimation of the disparity of the viewing vectors is to use scatter matrices.

The scatter matrix H of a vector set $\{w_i\}$ is defined as :

$$H = \sum_{w_i \in W_\mu(n)} w_i \, w_i^T \tag{19}$$

We may define the disparity of the set $\{w_i\}$ as the largest eigenvalue of H.

A.8   Experimental Results

The search algorithm with the combined geometric-probabilistic cost function is tested here for recognition capability and for differeniation power. The test is performed on 4 aeroplane models (a SAAB, a KFIR, a F15 and MIG25 - see Fig. 2A-2D respectively) and an industrial item (Fig.3). Both kinds of objects have specific points and straight edges primitives. These are segmented from their true images by means of preprocessing procedure that is described next. Each of the object's image regions is segmented from its digitized picture by means of compass edge operators that retain also the edge directional information. This information is utilized by the thinning and bridging procedures [11] that follow. An edge segmentation program [11] based on local curvature finds the corners, straight lines, and arcs of the edges. The contour and the plane's largest blobs are also founds by a graph-following-procedure. The specific points and straight edges selected are those which satisfy a set of topological properties [11]. The computation time on a VAX 11-750 computer,for preprocessing a typical 512 x 512 image was about 36 minutes (the programs were not optimized and no hardware convolvers were used). The experimental results of the search are summarized in Table 2. Each of the planes models and images are matched one against the other to test the recognition and differentiation power of the matching method. The rows of Table 2 relate to the images of the 4 planes and the columns to their models. In Table 2 one can also find the number of node examinations N_e required to reach the optimal match, number of matching errors M_e, (i.e., number of wrong labelings, such a assigning a wing tip of a model to a nose in the image) and the final cost function $C(n_t)$. The notation "exp" which appears in some of the cases instead of a numerical value means that the cost rised to values that indicated that the search will terminate only at very high values of Ne. Such cases occur when one tries to match totally incompatible model / image pairs. Though, the time (or node examinations)

required to detect incompatible matching is relatively short due to the steep rise in the cost function. The geometric cost function used here is based on area ratios. To examine the relative efficiency of the Euler angles geometric cost function; the industrial part (see Fig. 3) image is segmented to 13 straight edges and 14 specific points and is matched to its model using both geometric cost functions. The number of node examinations required to reach the final match with the areas ratio cost was 1331. The number of nodes examined with the Euler angles was 1821. (No search speeding up procedures [11] were employed.)

A.9 Discussion

The experimental results of the search based recognition method, demonstrate a significant reduction in the number of node examinations that are required to reach the optimal matching.

Matching of complex objects with the order of 12-18 primitives that require the order of 10^{12}-10^{22} nodes examinations with exhaustive search procedures are reduced to the order of 10^3. In the present study only initial-search-labeling-probabilities were used [11]. Thus, the cost function is mainly geometric. Results presented in [11] show that the robustness of the recognition method can be increased by the addition of the full probabilistic information, while at this stage is is partial. The main effort of future research should concentrate on incorporation of more information sources.

Table 2 : Cross Matching Results of 4 Aeroplanes.

Images \ models	SAAB	KFIR	F15	MIG25
SAAB	$N_e = 491$ $M_e = 0$ $C(n_t) = 0.31$	$N_e = 1569$ $M_e = 4$ $C(n_t) = 0.71$	exp	exp
KFIR	$N_e = 1596$ $M_e = 5$ $C(n_t) = 0.67$	$N_e = 421$ $M_e = 1$ $C(n_t) = 0.25$	exp	exp
F15	exp	exp	$N_e = 631$ $M_e = 0$ $C(n_t) = 0.35$	$N_e = 1555$ $M_e = 4$ $C(n_t) = 0.72$
MIG25	exp	exp	$N_e = 1387$ $M_e = 4$ $C(n_t) = 0.68$	$N_e = 771$ $M_e = 1$ $C(n_t) = 0.38$

B.1   Angles and Distances

This section describes two novel probabilistic models of imaged angles and distances. The models are then employed for model based 3-D objects recognition. These models confirm two general rules which may also be perceived intuitively. The first rule is the following: "If the viewing orientation is apriori undetermined, there is high probability that values of image angles are close to their depicted 3-D scene angles". Under the same apriori viewing conditions, a similar rule applies to image and scene distances: "It is very probable that relatively short (or long) distances in the image depict relatively short (or long) distances of the viewed 3-D scene". In other words, closer points in the image are more likely to depict shorter distances in the scene and the ratio of scene angles to their projected angles in the image usually have values which are close to unity.

Quantitative illustrations and proofs of these rules are given in Sections B.2 and B.3. Both of these rules, are based on the assuption of general view point which grants isotropic probabilities to all the viewing orientations around the observed scene. To represent this assumption an "Observability Sphere" is constructed.

The observability sphere is a scene centered imaginary sphere of very large radius which is practically infinite (see Fig. 9 in Appendix A). Each point on the sphere's surface represents a viewing direction vector which is normal to the surface at that point and has equal observation probability density. By relatively simple integration procedures which are described in Sections B.2 and B.3, the general probability densities of imaging transformations of angles and distances and the above mentioned rules can be derived. As elaborated in appendix A, the observability sphere can be employed for the estimation of other useful parameters such as initial labeling and joint probabilities of various object features.

The angles that are formed by linear features of an object are used as reliable primitives in its recognition process. An angle is defined by the relative orientation of its arms. Thus an object's angles are defined by its linear features (not necessarily touching). The process of orientation estimation of these features in the image, is usually based on extraction of large sets of edge points. Therefore, angular data is less susceptible to noise and partial occultation than data based on more local features such as specific points [11], edge junctions, etc.

As related literature it is worthwhile to mention other works that used the concept of general view point and used angular information. Witkin [12] addressed the recovery of surface orientation from natural images of textured surfaces. The slant and tilt angles of planar and curved surfaces were estimated statistically from measurements of tangent angles to contours in the image. Witkin [12] used the Gaussian sphere to obtain a joint probability density function of the observed tangent directions. This function was then used to receive maximum likelihood estimation of the surface orientation. Although Witkin's probability density functions and the techniques he used for their computation are completely different from ours, the principle of isotropic viewing orientations is adapted also there.

The general view point concept was used also by kanade [15] who used it for heuristic rules of parallel lines and skewed symmetry. Stevens [16] did not use the concept of skewed symmetry, but he presented a good body of psychological experiments which suggests that human observers can perceive surface orientations from figures with this property. Thus, we note that the angular information is one of the principal information sources for skewed symmetry detection.

Angular information was also used by Augusteijn and Dyer [17] for recovering 3-D planar surface orientation from a single 2-D polygonal contour of point pattern. This was done by iterative recovery of the slant and tilt angles of the pattern plane.

In this paper we address the problem of model based 3-D object recognition. In such problems the object and its orientation are usually apriory unknown. In the process of model based recognition it is required to match the model features to the equivalent features in the image. In the case of angles as features, most of the 3-D model's angles values do not resemble the observed 2-D image angles. This effects is due to the projection process which alters the image angles considerably. Using the orthographic projection approximation to the actual central projection, the image angles are non-linear functions of the slant and tilt of the 3-D angle plane relative to the image plane. Thus, one of the main obstacles to the matching algorithm using angles as features are the apriory unkown relations between the image and model angles. A quantitative formulation of these relations is given in section B. 2 .

A solution to the matching problem is proposed in this paper by employing the above mentioned probability model of angles transformation as an estimation criterion for the image-model angles relations. The matching algorithm is performed by the stochastic labeling algorithm [13]. This algorithm, updates in parallel a set of vector labeling probabilities of the image primitives. The relations between the image and model primitives are represented by a set of compatibility coefficients and are employed extensively by the stochastic labeling algorithm. The coefficients are evaluated from the above mentioned probability models of angles and distances. The methods for compatibility coefficients evaluation are described in [8]. Reference [8] describes the experiments of 3-D object recognition with stochastic labeling based on the probability models that are derived in Sections B.2 and B.3. As demonstrated in [8], the experimental recognition of synthetic or real objects yields satisfactory results.

B.2   The probability distribution model of projected angles

The computation of the orthographic projection of a 3-D scene angle denoted by α to an image angle denoted by β is simple [8]. Hence, $\beta = \beta(\alpha, \sigma, \tau_0)$ is a function of three parameters: the model angle α, the slant σ and the tilt of the bisector of α denoted by τ_0.

$$\beta(\alpha,\ \sigma,\ \tau_0) = \text{arctg}[\frac{\cos\sigma\,\text{tg}\,\alpha}{1 - 0.5\sin^2\sigma(1 - \frac{\cos 2\tau_0}{\cos\alpha})}] \tag{20}$$

The parametric behaviour of β as a function of α and τ_0 is given for example, in Fig. 4 (for $\alpha = 45^\circ$). From Fig. 4 it is obvious that the angle β variations as a function of the tilt τ_0 are larger when the slant σ is larger.

The observability sphere which represents the assumption of isotrophism of the viewing orientations is used here for the computation of the probability density of the angle ratio β/α. Referring to Fig. 5, the plane of the angle α coincides with the equator plane T of the observability sphere of radius R.

Each point on the sphere represents a viewing orientation denoted here by the viewing vector v. The imaging plane W is normal to v. The equator plane T is slanted by the angle σ and tilted by the angle τ relative to the imaging plane W. The infinitesimal observation probability of a certain (σ,τ) pair with differentials $d\sigma$, $d\tau$ is proportional to an area element dA on the observability sphere surface. The area element size varies according to:

$$dA = 2\pi R^2 \sin\sigma \, d\sigma \, d\tau \qquad (21)$$

The observation probability of dA is Δp, where

$$\Delta p = \frac{1}{2} \sin d\sigma \, d\tau \qquad (22)$$

A logarithmic scale is preferred for the angle ratio β/α so as to reflect the symmetry of the ratios probability density. The density is calculated by method similar to histogram calculation. The range of $\log(b/a)$ is divided into N odd number of intervals D_i; $i = 1,...,N$; by:

$$\frac{4(i-1)}{N-2} - 2 \leq \log(\beta/\alpha) \leq \frac{4i}{N-2} - 2 \qquad (23)$$

with end intervals defined by:

$$D_1: \quad -\infty < \log(\beta/\alpha) \leq -2$$

$$D_N: 2 \leq \log(\beta/\alpha) < \infty \qquad (24)$$

An integrator is associated with each interval D_i and a Δp is added to it whenever $\log(\beta/\alpha)$ belongs to D_i.

The curves depicted in Fig. 6 describe in logarithmic scale (with dashed lines) the probability density of the ratio β/α for acute $(0 < \alpha < \pi/2)$, obtuse $(\pi/2 \leq \alpha < \pi)$ angles, and the full a angle range (with solid line). The conditional probability distribution $p(\beta/\alpha)$ is illustrated by two dimensional drawing in Fig. 7. To test the stability of the results, the integrations are

pergformed with various α sets (0.5° to 5° intervals) and the various differentials $\Delta\tau$, $\Delta\sigma$. The conclusion that the densities are stable when $\Delta\sigma \rightarrow 0$, $\Delta\tau \rightarrow 0$ and $\Delta\alpha \rightarrow 0$ is based on the results obtained with various $\Delta\alpha$, $\Delta\tau$, $\Delta\sigma$ parameters. Below 5° there was no noticeable change in the densities.

The sharp peaks of the densities in Fig. 6 at $\log(\beta/\alpha) = 0$ verify the first rule presented in the introduction. This rule claims that there is a high probability that the values α and β are close. The probability, for example, of $|\log(\beta/\alpha)| \leq 0.3$ is larger than 0.84!

B.3 The probability Density of Projected Distances

A similar technique that was used to estimate the probability density of the projected angles in Section B.2 is employed here to compute the probability density of projecteddestances. The orthographic projection is also used here to approximate the actual central projection. By this approximation a vector **a** in the 3-D scene is projected to a vector **b** on the imaging plane by the equation:

$$b = s[a-v(a \cdot v)] \tag{25}$$

where the viewing unit vector is denoted by **v** and the imaging scale factor by s.

Since s is the same for all the viewing orientations (from the same distance) is is not taken into account. Theprobability density of the projection ratio $r = |b| / |a|$ is described in Fig. 8. The peak at $r = 1$ substantiates the second statement in the introduction. For instance, the probability for r to be in the range of $0.5 \leq r \leq 1$ is above 86 percent. This result also shows that apriori there is a high chance that the ratio of two distances in the image is close to the ratio of their respective distances in the 3-D scene.

B.4 Summary

We discussed in this paper the use of simple features for 3-D object recognition. The main advantage of these features over larger ones is their relatively easy and reliable segmentation. From our matching experiments we found out that in most cases the representation of rigid objects by simple features is sufficient for unique recognition.

The probabilistic models of angles and distances supply strong cues for the grouping processes of the above mentioned features. Recently we found that also the probability density of curvature has similar properties to angles and distances. These results will be reported shortly.

Appendix A: The Observability Sphere and Its Uses

In this appendix we describe the observability sphere and its uses. The sphere enables the estimation of the observation probabilities of various object features. Joint observation probabilities can also be computed by using the sphere. These probabilities have two uses: they can serve as initial labeling probabilities, and participate in the estimation of the compatibility coefficients needed for stochastic labeling. The observability sphere concept was also used for the computation of the probability densities of angles and distances as described in sections B.2 and B.3.

Referring to Fig. 9, the observability sphere is an imaginary sphere with very large radius which is practically infinite. The observed scene or object are placed approximately at the sphere's center. The sphere's radius is made very large so that the exact location of the center will not influence the viewing orientation of the object features. Each point on the sphere's surface represents a viewing orientation vector which is normal to the sphere's surface at that point. Actually we represent by the sphere the assumption of apriori isotropic probability of viewing orientation around the observed object or scene. For that reason the sphere is defined to have constant observation probability density on its surface points. This assumption may be adjusted for objects which have parts that are occluded permanently.

The apriori observation probability of any feature that belongs to the observed object is proportional to the area of the viewing region on the observability sphere's surface. The viewing region of a feature is the union of all the points on the surface representing viewing orientations from which that feature can be observed.

To find the viewing region of a certain feature one has to perform a central projection of that feature on the sphere's surface. This is done by locating all the valid centers of projection and emitting rays from them to sphere's surface in all directions which are not occluded. For that purpose, the set of valid centers of projection is the set of feature's points. This projection is not simple to perform unless one is dealing with features that are planar, i.e. contained within a plane. Planar features are features such as planar faces, straight edges or specific points [7], [11]. The central projection of such features is relatively simple. For example a central projection of a cube's face denoted by "a" in Fig. 9 is the region F(a). This region is almost a hemisphere when the cube's edge size 'd" is very small compared to the sphere's radius R:

$$F(a) = 2\pi R^2 - (d/2)2\pi R \to 2\pi R^2 \qquad \text{for} \qquad d << R \qquad (A.1)$$

Thus, the observation probability P(a) of that face a is very close to 0.5:

$$P(a) = F(a) / 4\pi R^2 \cong 1/2 \qquad (A.2)$$

Now, let's assume a convex polyhedral object with planar faces $a_1,...,a_n$ with respective normals $t_1,...,t_n$. Each viewing region $F(a_m)$ of a face a_m is close to a hemisphere. The probability for simulaneous observation of the group $a_1,...,a_m$ denoted by $P(a_1,...,a_m)$ is proportional to the *intersection* of their respective viewing regions, that is:

$$P(a_1, ..., a_m) = \frac{1}{4\pi R^2} \bigcap_{j=1}^{m} F(a_j) = \frac{1}{4\pi R^2} \iint \prod_{j=1}^{m} A(v(s) \cdot t_j) ds$$

(A. 3)

$$A(v(s) \cdot t_j) = \begin{cases} 1 & \text{if} & v \cdot t_j > 0 \\ 0 & \text{if} & v \cdot t_j \leq 0 \end{cases}$$

where s is a differential area element of the sphere and v(s) is its observation vector. For example, if a_1, a_2 and a_3 are neighboring faces of a cube than the simultaneous observation probability $P(a_1,a_2)$ will be 1/4 and $P(a_1,a_2, a_3)$ is equal to 1/8.

Observation regions of other planar features are created by the operation of union. A straight edge is usually created by two neighboring planar faces. Thus, the observation region of that edge is the union of its bordering faces regions. We now denote the observation probability of a feature created by a union of $a_1,...,a_m$ by $P'_1(a_1,...,a_m)$. In the general case $P'_1(a_1,...,a_m)$ is given by:

$$P'(a_1, ..., a_m) = \frac{1}{4\pi R^2} \bigcup_{j=1}^{m} F(a_j) = \frac{1}{4\pi R^2} \iint \sum_{j=1}^{m} A(v(s) \cdot t_j) ds$$

(A. 4)

$$A(v(s) \cdot t_j) = \begin{cases} 1 & \text{if any of } v \cdot t_j > 0 ; \quad j = 1, ..., m; \\ 0 & \text{otherwise} \end{cases}$$

For instance, the observation probability of a cube's edge is $P'_1(a_1,a_2) = 3/4$ and of a cube's vertex $P'_1(a_1,a_2,a_3) = 7/8$. By a combination of the disjunction and conjunction operations the simultaneous observation probabilities of various features can be computed too [11]. This technique is also applicable to features of objects which are non-convex and curved [11].

227

Another use of the observability sphere is linked to the construction of Aspect Graphs. In fact, the probabilities of observation of various views, or aspects, of an object are not equal in general.

These probabilities can be computed easily using eq. (A.3). The observation probability of a certain node can be used as its weight in the general graph. By this way, many nodes with small weights may be pruned.

For example, the observation probabilities of a cube aspects no. 1 and 3, in fig. A.2, are close to zero (using orthographic projection), and aspect no. 2 has a probability of 1/8. Refering to fig. A.3 , the six faces of the cube are labelled as a,b,c and their respective opposites as $\bar{a}, \bar{b}, \bar{c}$.

Each aspect is labelled by a code of three letters indicating its visible faces. Some of the viewing regions on the sphere are indicated in fig. A.3. The pruned aspect graph of the cube is shown in fig. A.4.

References

1) R.A. Brooks, "Model Based 3-D Interpretations of 2-D Images", IEEE PAMI-5, No. 2, March 1983.

2) Akinniyi, F.A. and Wong, K.C., "A New Product Graph Based Algorithm For Subgraph Isomorphism", IEEE Proc. CVPR, Washington, June 1983.

3) You, M. and Wong, K.C., "An Algorithm For Graph Optimal Isomorphism", Proc. IJCPR, Montreal, Canada, Aug. 1984, p. 316.

4) Mulgaonkar, P.G., Shapiro, L.G. and Haralick, R.M., "Matching 'Sticks, Plates and Blobs' Objects Using Geometrical and Relational Constraints", Image and Vision Computing, Vol. 2, No. 2, May 1984, pp. 85-98.

5) R.M. Haralick, "Using Perspective ;Transformations in Scene Analysis", CGIP, Vol. 13, pp. 191-221, 1980.

6) D.P. Huttenlocher, S. Ullman, "Object Recognition Using Alignment", Proc. Darpas Image Understanding Workshop LA, Feb. 1987.

7) Ben-Arie, J. and Meiri, A.Z.: "3-D Object Recognition by Optimal Matching Search of Multinary Relations Graphs", Computer Vision, Graphics and Image Processing, Vol. 37, No. 3, March 1987, pp. 345-361.

8) Ben-Arie, J.: "Probabilistic Models of Viewed Angles and Distances With Application to 3-D Object Recognition", The Technion Aeronautical Eng. Dept. Technical Report: TAE618, Feb. 88.

9) Ben-Arie, J. and Meiri, Z.A.: "3-D Object Recognition By State Space Search: Optimal Geometric Matching", Proceedings of IEEE Computer Society Conf. on Computer Vision and Image Processing, Miami, Florida, U.S.A. June 1986, pp. 457-462.

10) Ben-Arie, J. and Meiri, Z.A.: "Optimal Recognition of 3-D Objects By Search: Generic Objects", Proceedings of IAPR 8th International Conf. on Pattern Recognition, Paris, France Oct. 1986, pp. 100-104.

11) Ben-Arie, J.: "Computer vision: 3-D Object Recognition From 2-D Single Images, Research Report, Dept. of Aeronautical Eng., Technion - I.I.T., July 1986, (in Hebrew).

12) A.P. Witkin, "Recovering Surface Shape and Orientation from Texture", Artificial Intelligence 17, 1981, pp. 17-47.

13) O.D. Faugeras, M. Berthod, "Improving Consistency and Reducing Ambiguity in Stochastic Labeling, An Optimization Approach", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-3, No. 4, July 1981, pp. 412-424.

14) S. Peleg, "A New Probabilistic Relaxation Scheme", IEEE, Vol. PAMI-2, No. 4, July 1980.

15) T. Kanade, "Recovery of the Three-Dimensional Shape of an Object From a Single View", Artificial Intelligence 17, pp. 409-460, 1981.

References (Cont.)

16) K.A. Stevens, "Representing and Analyzing Surface Orientation", in "Artificial Intelligence: A MIT Perspective", Vol. 2, PH.H. Winston and R.H. Brown (Eds.), Cambridge, MA, MIT Press, 1979.

17) M.F. Augusteijn, C.R. Dyer, "Model Based Shape from Contour and Point Patterns", IEEE Conf. of Computer Vision and Pattern Recognition, San Francisco, June 1985, pp. 100-105.

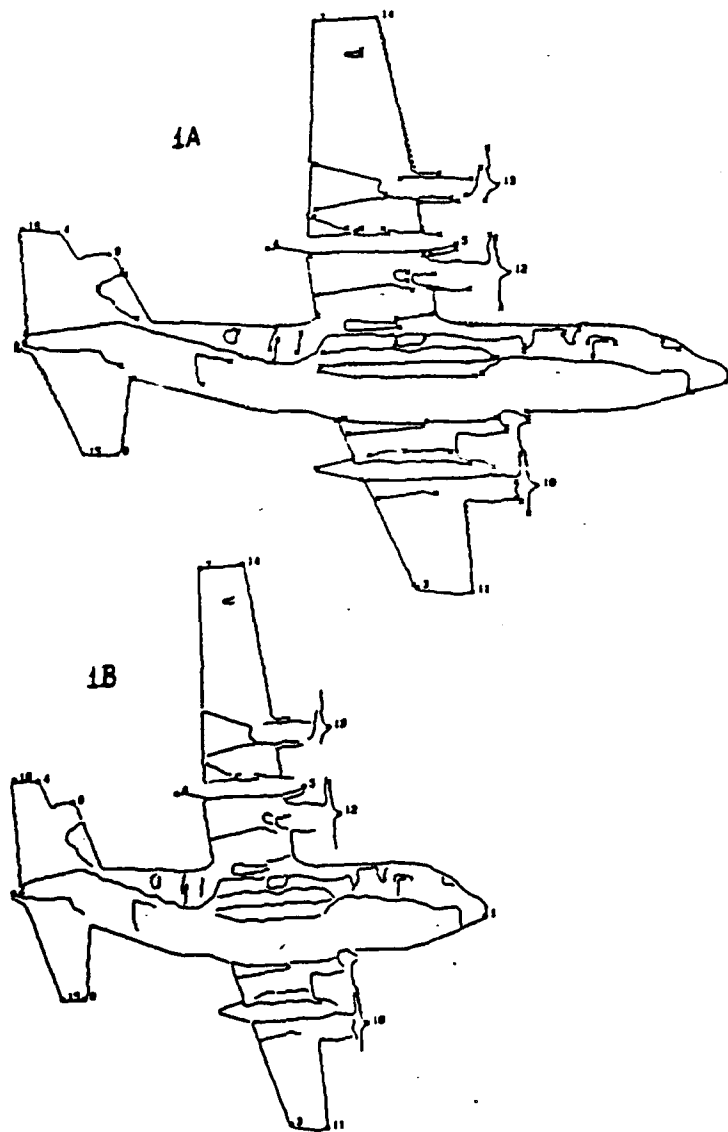Fig. 1: Segmented images of Hercules aeroplanes:
1A: Original image; 1B: Distorted image.

Fig. 2A-2D: Segmented images of 4 aeroplanes.
Specific points are denoted by
numbers, edge terminators by +, and
straight edges approximations by
dashed lines.

Fig. 2A: SAAB
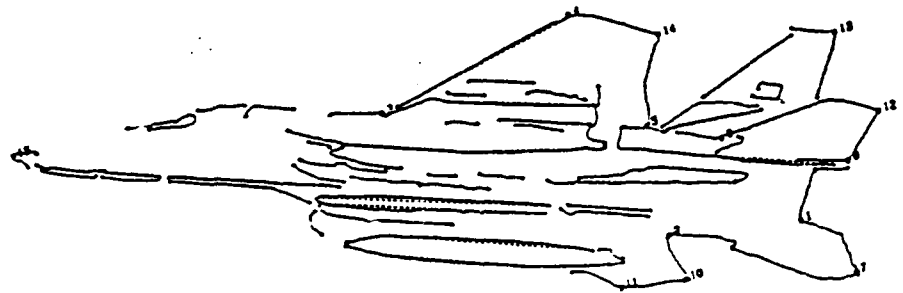
Fig. 2C: F15
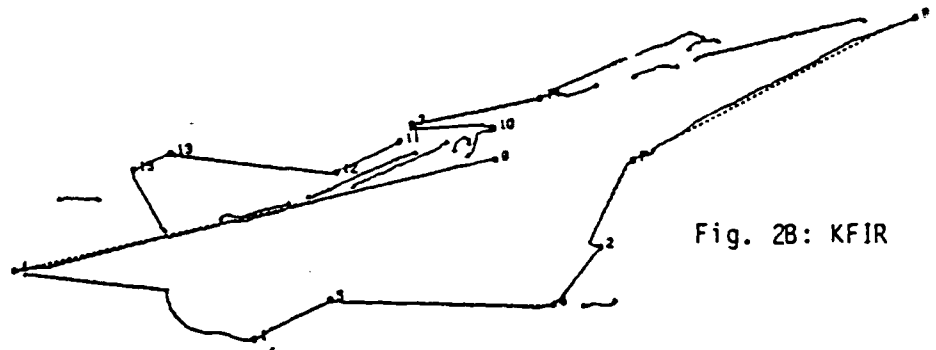
Fig. 2B: KFIR

Fig. 2D: MIG25
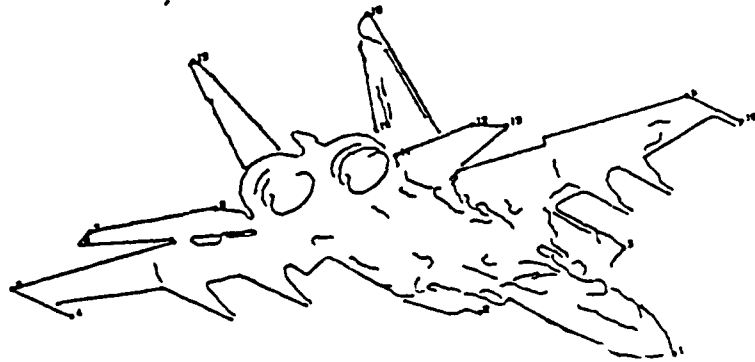
Fig.1c: Imaging parameters computation by edge
orientations.
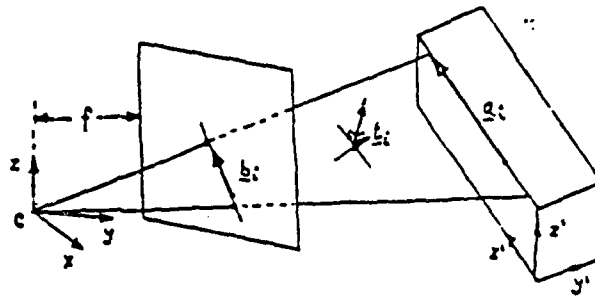
Fig. 3: Segmented image of an industrial item.
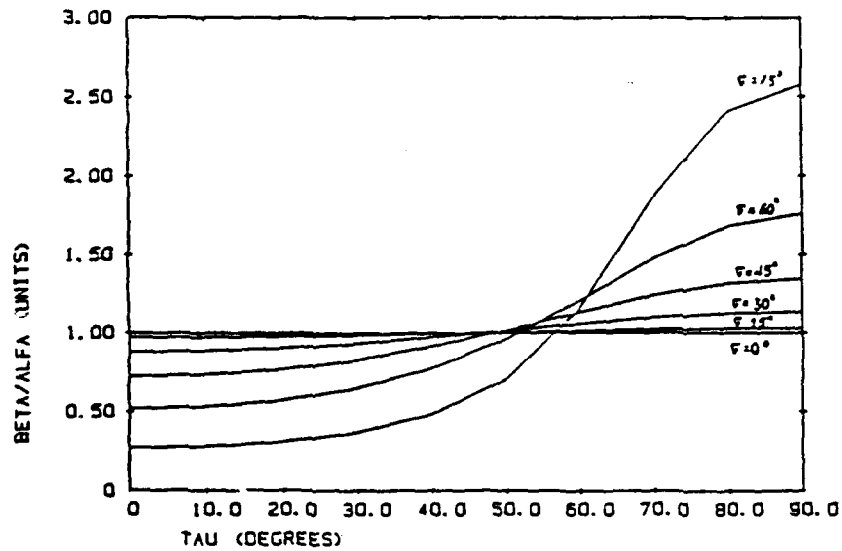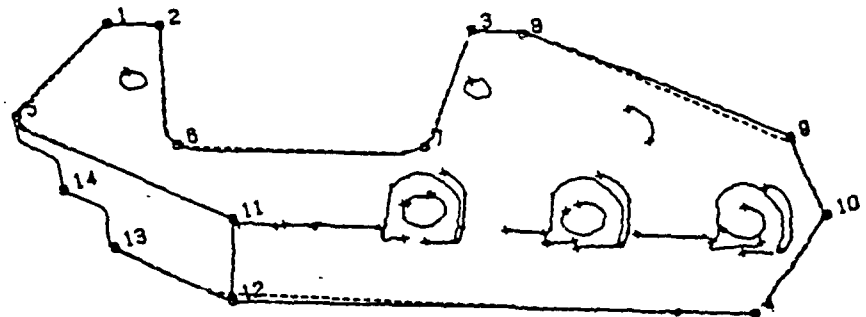
Fig. 4 : The parametric behaviour of the projected angle β as a function of
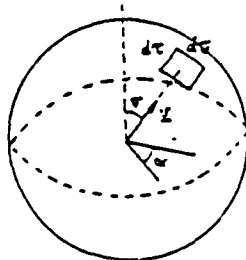the tilt τ_0 and the slant σ (for $\alpha=45°$).

Fig. 5 : The use of observability sphere for the computation of the
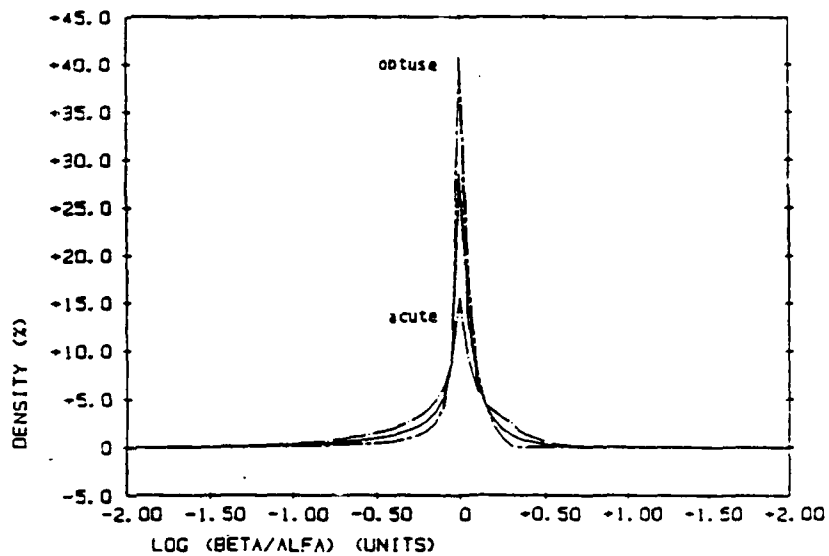probabilistic model of projected angles.

Fig. 6 : The probability density of the ratio a/a (in logarithmic scale).
For acute 0<a<π/2 and obtuse angles the density is illustrated by
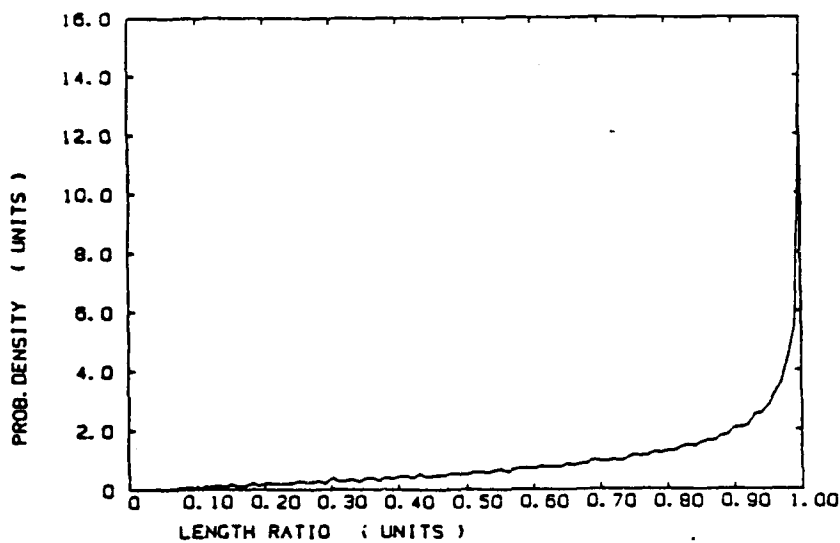dashed lines. The full a angle range is described by a solid line.

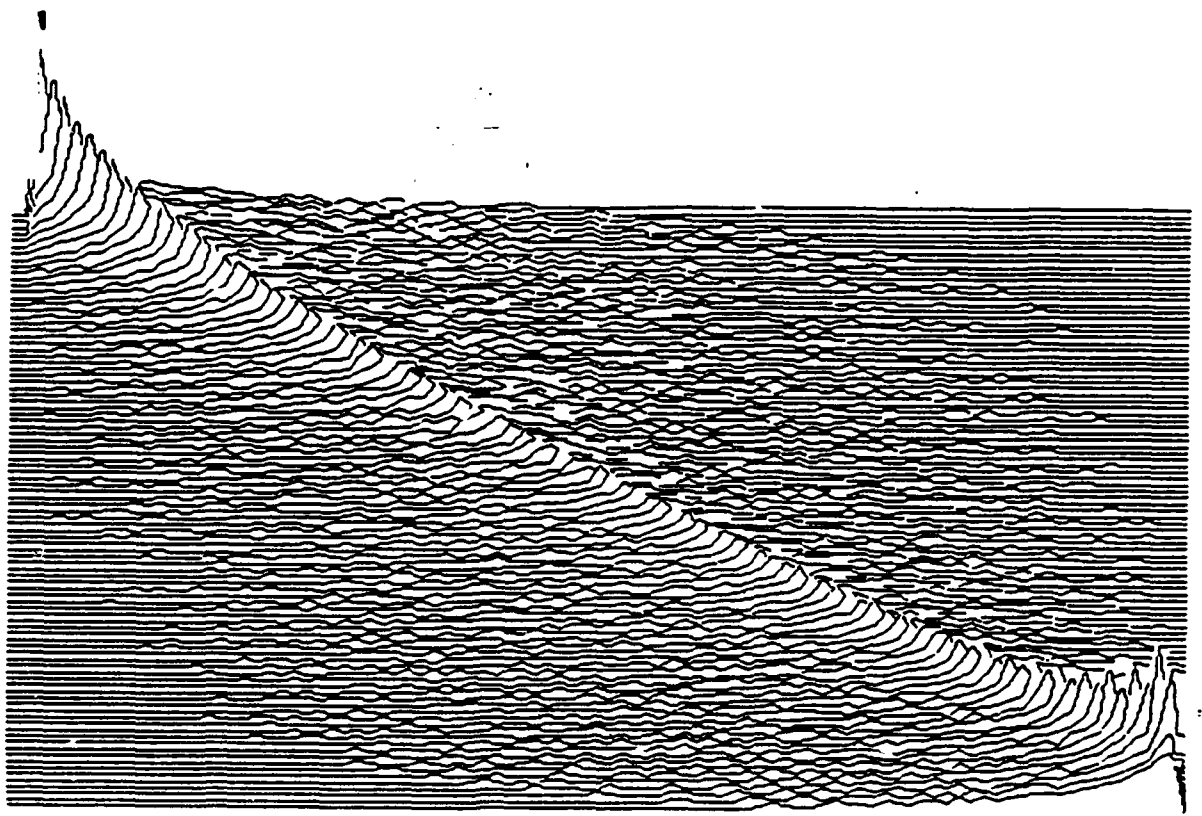
Fig. 8 : The probability density of ratios of distances.

Fig. 7 : The conditional probability distribution p(β|α). β is described by
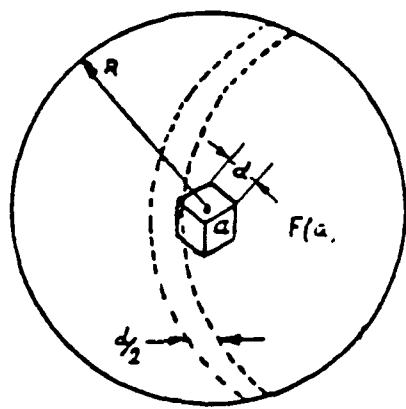the horizontal axis. Both angles are within the range: $3°\leq\alpha, \beta\leq177°$.

Fig. : Observation probabilities of object features computed by using the
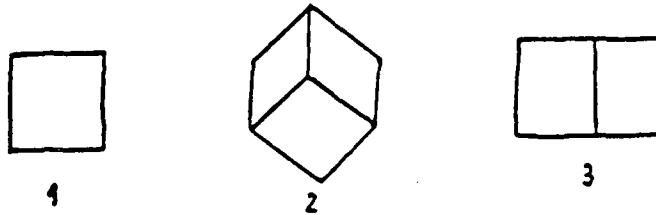observability sphere.
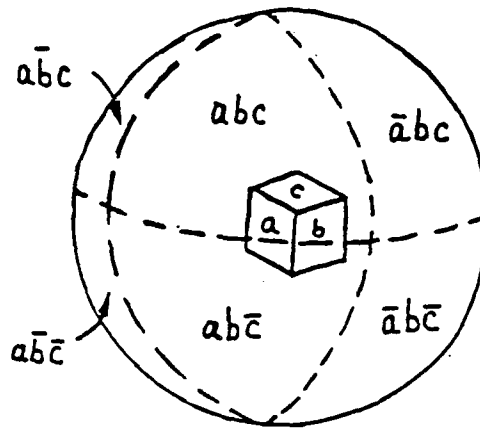
Fig. A.2 : The Three Aspects of a Cube.

Fig. A.3 : The Viewing Regions of a Cube's Aspects Indicated on the Observability Sphere.
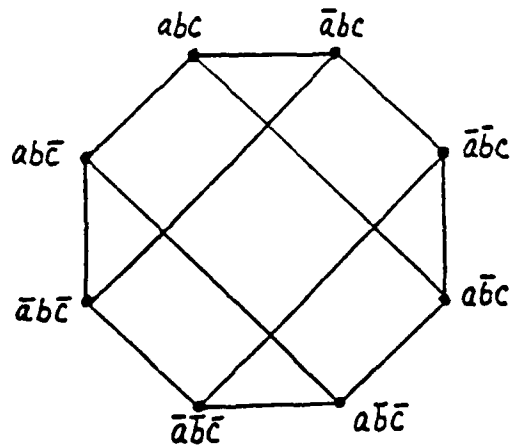

Fig. A.4 : The pruned Aspect Graph of a Cube.

236

Integrating Planning and Reactive Control[*]

Stanley J. Rosenschein
Leslie Pack Kaelbling
Teleos Research
576 Middlefield Road
Palo Alto, CA 94301

1 Introduction

Artificial intelligence research on planning is concerned with designing control systems that choose actions by manipulating explicit descriptions of the world state, the goal to be achieved, and the effects of elementary operations available to the system. Because planning shifts much of the burden of reasoning to the machine, it holds great appeal as a high-level programming method [3,10,12]. Experience shows, however, that it cannot be used indiscriminately because even moderately rich languages for describing goals, states, and the elementary operators lead to computational inefficiencies that render the approach unsuitable for realistic applications. This inadequacy has spawned a recent wave of research on "reactive control" or "situated activity" in which control systems are modeled as reacting directly to the current situation rather than as reasoning about the future effects of alternative action sequences [2,1,11]. While this research has confronted the issue of run-time tractability head on, in many cases it has done so by sacrificing the advantages of declarative planning techniques.

This paper discusses ways in which the two approaches can be unified. We begin by modeling reactive control systems as state machines that map a stream of sensory inputs to a stream of control outputs. These machines can be decomposed into two continuously active subsystems: the planner and the execution module. The planner computes a "plan," which can be seen as a set of bits that control the behavior of the execution module. An important element of this work is the formulation of a precise semantic interpretation for the inputs and outputs of the planning system. We show that the distinction between planned and reactive behavior is largely in the eye of the beholder: Systems that seem to compute explicit plans can be redescribed in situation-action terms and vice versa. We also discuss practical programming techniques that allow the advantages of declarative programming and guaranteed reactive response to be achieved simultaneously.

2 Planning and Reactive Control

Classical AI views the generation of behavior as a two-step process consisting of planning and execution. Planning produces a data structure describing a course of action; execution is the step-by-step interpretation of this data structure to produce overt behavior. The planning step can be viewed as a form of stylized program synthesis in a weak logic of programs, and many formalisms have been proposed to capture the logic of planning. A common approach is to employ predicate calculus formulas as state descriptions (e.g., *on(blockA,blockB)*) and to model operators as state-transforming functions, described either axiomatically (using facts of the form $holds(p,s) \rightarrow holds(q, op(s))$) or as syntactic transformations that map state descriptions to state descriptions. Letting *ops*, *init*, and *goal* stand for formulas expressing, respectively, facts about the operators, the initial conditions, and the goal statement, we require the planner to find *plan* = make-plan(*ops*, *init*, *goal*) such that

$$ops \models init \wedge plan \rightarrow goal \ .$$

In other words, it should follow from the operator descriptions that if the initial condition holds and the plan is carried out, the goal condition will be achieved. Note also that $init \wedge plan$ should be consistent; otherwise, the requirement can be trivially satisfied.

The complexity of plan synthesis obviously depends on the specific nature of the domain. For realistic domains, however, traditional planning typically requires significantly more time than the fundamental reflex cycle of the system, and controlling the rate at which planning occurs relative to changes in the environment is extremely challenging. For this reason, classical planning techniques have almost always been applied, in practice, to "static" domains, in which the only significant source of change is the agent itself and in which, therefore, the time required for planning can be safely ignored.

In an attempt to deal with more dynamic domains, some researchers have abandoned planning in favor of *reactive control*, which does not take a two-stage view of behavior generation. In this approach, the behavior of the agent is specified directly using situation-action rules that are evaluated at frequent intervals. A reactive control system could be implemented, for example, as a program executing a tight loop, the body of which exhibits a high degree of conditionality, for example:

```
do forever
      if tiger_approaching then
          set wheel velocities to [+30,+30],
      else if ...
```

Since the conditions can be evaluated in parallel, reactive systems can also be described as *circuits or operator networks* implementing a function that maps a stream of information states to a stream of output commands to the effectors. The key to reactivity is to design this function so that it can be computed quickly again and again.

Each approach has its advantages. Planning provides a convenient high-level declarative formalism and leaves much of the reasoning to the machine. In principle, this makes it possible for the control system to handle classes of situations that are too complex for the programmer to anticipate in advance but are amenable to analysis at run time, once a concrete initial state and goal state are available. In contrast, reactive control offers the advantage of guaranteed

response time and hence the ability to react quickly to a changing environment. Because neither approach clearly dominates the other and because many application domains have attributes that make each attractive, a synthesis of these two techniques is necessary.

One method for achieving such a synthesis is to embed a reactive controller in a classical planner-based architecture. In a sense, this is what the term "execution monitoring" is often taken to mean in classical planning: The planner sends a data structure to the execution module, which in turn reacts to changing world conditions under the control of the plan. The execution module is also able to detect conditions in the world that violate the assumptions upon which the plan's correctness depends. Unfortunately, the mathematical framework of classical planning, based on atemporal state transformations, offers little guidance as to how the passage of time during the planning process ought to be handled.

Since reactive control is based on a model of time-bounded computation, it is more natural to incorporate planning by extending the reactive-control architecture rather than vice versa, and this is the approach we shall take. In order to do this, however. we must first characterize the semantics of the data structures produced by the planner in a way that makes sense in the reactive control model.

3   Semantics for Planning and Control

We shall model a control system as a state machine that transduces inputs carrying information about the environment to outputs that affect the environment. In the simplest case, this machine has no state and simply computes a pure function from inputs to outputs. In more complex cases, including cases in which significant planning occurs, the computation requires internal state. A major challenge in designing control systems is to provide a clear semantic model of the information available to the control system, of the goals achieved by the chosen actions, and of the mapping between the two.

Let M be a control system with input variable in, output variable out, and an internal state vector a. The inputs carry information about the world, the outputs are commands to the effectors, and the internal state allows the computation of outputs to depend on past inputs and to be extended in time. To introduce a planner into this model, we decompose the mac_ine into components, introducing three subsidiary variables, $init$, $goal$, and $plan$, and four sub-machines: E_{init}, E_{goal}, $Planner$, and $Exec$. We assume that ops is fixed in advance. The inputs and outputs of these modules are as follows:

- E_{init}: input in, output $init$

- E_{goal}: input in, output $goal$

- $Planner$: input $init$, $goal$, output $plan$

- $Exec$: input in, $plan$, output out

The overall structure of the machine is illustrated in Figure 1. Informally, the E_{init} and E_{goal} machines operate on the input, extracting values representing the initial conditions and goal condition, respectively. These are transduced by $Planner$, in a way that may involve internal state and computation over time, to a continuously available $plan$ output. Note, however,
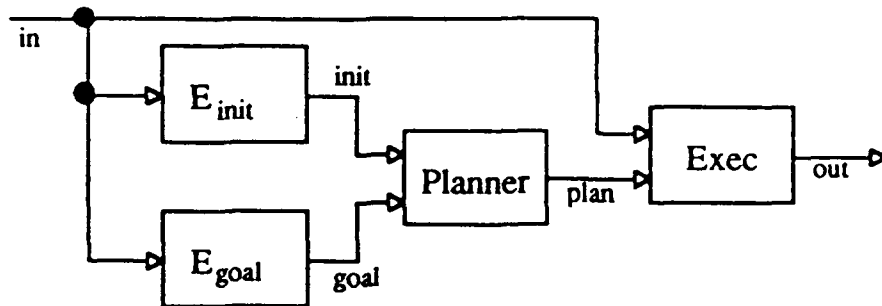
Figure 1: Embedding a planner in a reactive control system.

that the output may be vacuous, indicating that the final plan has not yet been computed [4].

We are interested in characterizing the semantics of the inputs of *Planner* and of its result, but must first consider the more general question of where semantic "interpretations" for data values come from.

For data structures like *init*, the classical view is that the data value is a description of facts about the world expressed in some language whose semantics is clear to the designer of the system. This description would be of little use were it not also the case that when the data structure had a particular value, the condition denoted was guaranteed to hold in the environment. Such semantic considerations form the foundation of the situated-automata model in which the semantics of data structures are characterized in terms of objective correlations with external reality rather than in terms of designer-stipulated interpretations. In this approach, one says a machine variable x carries the information that p in world state s, written $s \models K(x,p)$, if for all world states in which x has the same value it does in s, the proposition p is true. The formal properties of this model and its usefulness for programming embedded systems have been described elsewhere [7,8,5,9].

Since we are committed to an information-based semantics for reactive systems, we seek an "objective" semantics of goals defined explicitly in informational terms. We can reformulate the notion of having a goal p as having the information that p implies a fixed top-level goal, called N for "Nirvana." Formally, we define a goal operator G as follows:

$$G(x,p) \equiv K(x,p \rightarrow N)$$

In this model, x has the goal p if x carries the information that p implies Nirvana.[1] Since this defines goals explicitly in terms of information, the same formal tools used to study information can be applied to goals as well. In fact, under this definition, goals and information are dual concepts.

To see this, consider a function f mapping values of one variable, a, to values of another variable, b. Under the information interpretation, such a function takes elements having more specific information into elements having less specific information. This is because functions generally introduce ambiguity by mapping distinct inputs to the same output. For example, if value u_1 at a is correlated with proposition p and value u_2 at a is correlated with q and if

---

[1] We observe that under this definition *False* will always be a goal; in practice, however, we are only interested in non-trivial goals.

f maps both u_1 and u_2 to v at b, the value v is ambiguous as to whether it arose from u_1 or u_2, and hence the information it contains is the disjunctive information $p \vee q$, which is less specific than the information contained in either u_1 or u_2. Thus, functional mappings are a form of forgetting.

Under the goal interpretation, this picture is reversed. The analog to "forgetting" is committing to subgoals, which can be thought of as "forgetting" that there are other ways of achieving the condition. For instance, let the objective information at variable a be that the agent is hungry and that there is a sandwich in the right drawer and an apple in the left. If the application of a many-to-one function results in variable b's having a value compatible with the agent's being hungry and there being a sandwich in the right drawer and either an apple in the left drawer or not, we could describe this state of affairs by saying that variable b has lost the information that opening the left drawer would be a way of finding food. Alternatively, we could say that variable b had committed to the subgoal of opening the right drawer. The phenomena of forgetting and commitment are two sides of the same coin.

Formally we can relate this observation to axioms describing information and goals. One of the formal properties satisfied by K is the deductive closure axiom, which can be written as follows:

$$K(x, p \to q) \to (K(x, p) \to K(x, q)) \ .$$

The analogous axiom for goals is

$$K(x, p \to q) \to (G(x, q) \to G(x, p)) \ .$$

This is precisely the subgoaling axiom. If the agent has q as a goal and carries the information that q is implied by some other, more specific, condition, p, the agent is justified in adopting p as a goal. The validity of this axiom can be established directly from the definition of G.

Given these two ways of viewing the semantics of data structures, we can revisit the *Planner* module with inputs *init* and *goal* and output *plan*. The most natural way to interpret the values of these variables is to apply the information interpretation to the values of *init* and the goal interpretation to the values of *goal* and *plan*. However, as observed above, since the goal interpretation is derived directly from the informational model, we could have applied either interpretation to any of the values.

In summary, one need not think of "planning" as an essentially different kind of function performed by the system. Rather, it can be thought of as a *perspective* one takes on certain data structures when one thinks of them—for design convenience—as encoding goals rather than information.

4   Current Research Directions

In this section we list several efforts currently underway that are aimed at exploring the practical consequences of our approach toward integrating planning and reactive control.

4.1   Embedding Planning in Gapps

Gapps [6] is a declarative language for programming reactive systems. The Gapps compiler takes as input a top-level goal and a set of goal reduction rules and produces as output a

program for achieving the top-level goal. The program is guaranteed by construction to map information states to actions in constant time. By using Gapps, the programmer can gain many of the benefits of declarative programming without sacrificing real-time response. One direction of research is to embed planning in Gapps by converting operator descriptions into goal reduction rules, which in turn are transformed by Gapps into real-time programs. A typical rule schema might be:

```
(defgoalr (ach P)
   (if (regress P a)
       (do a)
       (ach (regress P a))))
```

Because Gapps produces a fixed-size circuit at compile time, a compile-time bound must be placed on the *depth* of the regression, although in principle the actual calculation of the regressed condition can be deferred to run time.

4.2   Temporally Extended Planning Processes

Traditional planners operate by carrying out a guided search through a space of plans. Depending on the combinatorics of the search, this process may or may not succeed within a single cycle of the reactive system. If it does not, the search must proceed in parallel with the execution of a more reactive, though perhaps less effective, behavior. Since the passage of time affects whether or not a data value will continue to be correlated with the environment, it is clear that the semantics of temporally-extended planning will be time-dependent. A simple solution to this problem is for the planner to monitor world conditions that would invalidate the current plan and to output the vacuous plan when those conditions arise [4]. While correct, this approach is not maximally information-preserving and more subtle methods are possible. In the case of informational data structures, we have explored declarative programming techniques to control the updating of the machine's information state so that maximal correlation with the environment is maintained [9], and similar methods might be applied to planning over time as well.

4.3   Trading Flexibility for Performance

As in conventional programming, some information required for action selection might be available at compile time, while other information may become available only at run time. Ease of programming would be enhanced by minimizing syntactic and semantic distinctions based only on differences as to when information becomes available. In traditional compilers, for instance, constant-folding optimizations take advantage of compile-time information about the values of expressions in a way that is entirely transparent to the programmer. For planning and control applications, this transparency is more difficult to achieve because without sufficient compile-time information, the symbolic synthesis procedure may not terminate, and without a clear compile-time versus run-time model in mind, the programmer may lack sufficient insight to adequately control the compilation process. Nevertheless, our ultimate goal is to make it as easy as possible to trade off flexibility against performance by conveniently moving the boundary between compile-time and run-time processing.

Acknowledgments

References

[1] Agre, Philip E. and David Chapman. "Pengi: An Implementation of a Theory of Activity." *Proceedings of the Sixth National Conference on Artificial Intelligence*, Seattle, Washington (July 1987).

[2] Brooks, Rodney A. "A Robust Layered Control System for a Mobile Robot." Technical Report 864, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts (1985).

[3] Fikes, Richard and Nils J. Nilsson. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving." *Artificial Intelligence*, Vol. 2, Nos. 3,4 (1971).

[4] Kaelbling, Leslie P. "An Architecture for Intelligent Reactive Systems." In Michael P. Georgeff and Amy L. Lansky, editors, *Reasoning about Actions and Plans: Proceedings of the 1986 Workshop*. Morgan Kaufmann, Los Altos, California (1987).

[5] Kaelbling, Leslie P. "Rex: A Symbolic Language for the Design and Parallel Implementation of Embedded Systems." *Proceedings of the AIAA Conference on Computers in Aerospace*, Wakefield, Massachusetts (1987).

[6] Kaelbling, Leslie P. "Goals as Parallel Program Specifications." *Proceedings of the Seventh National Conference on Artificial Intelligence*, Morgan Kaufmann, St. Paul, Minnesota (August 1988).

[7] Rosenschein, Stanley J. "Formal Theories of Knowledge in AI and Robotics". In *New Generation Computing*, Vol. 3, No. 4, (special issue on Knowledge Representation), Ohmsha, Ltd., Tokyo, Japan (1985).

[8] Rosenschein, Stanley J. and Leslie P. Kaelbling. "The Synthesis of Digital Machines with Provable Epistemic Properties," *Proceedings of Workshop on Theoretical Aspects of Reasoning About Knowledge*, Monterey, California (1986).

[9] Rosenschein, Stanley J. "Synthesizing Information-Tracking Automata from Environment Descriptions." *Proceedings of the First Conference on Principles of Knowledge Representation and Reasoning*, Toronto, Canada (to appear).

[10] Sacerdoti, Earl. *A Structure for Plans and Behavior*. Elsevier Noth-Holland, Inc., New York (1977).

[11] Schoppers, Marcel J. "Universal Plans for Reactive Robots in Unpredictable Environments." *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Morgan Kauffman, Milan (1987).

[12] Wilkins, David E. *Practical Planning: Extending the Classical AI Planning Paradigm.* Morgan Kaufmann Publishers, Inc. San Mateo, California (1988).

Distribution List for IDA Document D-649

NAME AND ADDRESS	NUMBER OF COPIES
Sponsor	
Lt Col Robert Simpson Defense Advanced Research Projects Agency 1400 Wilson Blvd. Arlington, VA 22209-2308	3
Others	
Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Prof. S. Ullman The Weizmann Institute of Science Rehovot, Israel	19
Dr. Saul Amarel c/o Elsie Jackson Chairman, Dept. of Computer Science Hill Center/Busch Campus Rutgers University New Brunswick, NJ 08540	1
Thomas O. Binford Computer Science Dept. Stanford University Sanford, CA 94305	1
Dr. Martin A. Fischler SRI International EK 290 333 Ravenswood Ave Menlo Park, CA 94025	1
Dr. Takeo Kanade CMU Computer Science Dept. Pittsburgh, PA 15213-3890	1
Dr. Richard Korf Computer Science Dept. UCLA 405 Hilgard Ave. Los Angeles, CA 90024	1

NAME AND ADDRESS	NUMBER OF COPIES
Dr. Edward M. Riseman U Mass COINS Dept. Lederle Bldg, GRC Amherst, MA 01003	1
Dr. Azriel Rosenfeld University of Maryland Center for Automation Research College Park, MD 20742	1
Dr. Stanley Rosenschein DJ 247 AI Center SRI International 333 Ravenswood Ave. Menlo Park, CA 94025	1

IDA

General W.Y. Smith, HQ	1
Mr. Philip L. Major, HQ	1
Dr. Robert E. Roberts, HQ	1
Ms. Anne Douville, CSED	1
Dr. John F. Kramer, CSED	1
Mr. Terry Mayfield, CSED	1
Dr. Richard Wexelblat, CSED	1
Mr. Michael Bloom, CSED	1
Ms. Helen Singleton, CSED	1
Ms. Sylvia Reynolds, CSED	2
IDA Control & Distribution Vault	2