LEVEL $\overline{II}$

437

AD A101215

DTIC
S ELECTE D
JUL 1 0 1981
E

rochester

Department of Computer Science
University of Rochester
Rochester, New York 14627

81 6 12 017

Parameter Networks:
Towards a Theory of Low-Level Vision

D.H. Ballard
Computer Science Department
University of Rochester
Rochester, NY 14627

TR 75
Apr 1981

JUL 1 1981

Abstract

One of the most fundamental problems in vision is segmentation; the way in which parts of an image are perceived as a meaningful whole.

Recent work has shown how to calculate images of physical parameters from raw intensity data. Such images are known as intrinsic images, and examples are images of velocity (optical flow), surface orientation, occluding contour, and disparity. The principal difficulty with intrinsic images is that each by itself is generally underconstrained; they can only be computed in parallel with each other and with the use of parameters obtained through segmentation.

While intrinsic images are not segmented, they are distinctly easier to segment than the original intensity image. If parts of these images are organized in some way, this organization can be detected by a general Hough transform technique. Networks of feature parameters are appended to the intrinsic image organization. Then the intrinsic image points are mapped into these networks. This mapping will be many-to-one onto interesting parameter values. This basic relationship is extended into a general representation and control technique with the addition of three main ideas: abstraction levels; sequential search; and tight coupling. These ideas are a nucleus of a theory of low-level and intermediate-level vision. This theory explains segmentation in terms of highly parallel cooperative computation among intrinsic images and a set of parameter spaces at different levels of abstraction.

1. Overview

One of the most troublesome puzzles in vision is how parts of an image are seen as a meaningful whole or *segment*. This is known as the segmentation problem. The ambiguous use of segment, which means part, to denote a whole, arises from the fact that a segment is an intermediate component in a description which relates an object with an image. From the viewpoint of the object description, the segment is a part. From the viewpoint of a group of image points with common properties, the segment is a whole.

Parts of an image are seen as a segment if the corresponding physical object has common *physical properties*, or features. For example, if a connected component of the image has a single color, say red, then it may be seen as a segment. The patch of red arises from the physical object's surface reflectance. Usually there are not one but several features which have the same spatial registration. For example, an object may be moving, red, and a cube. Figure 1a shows this case. Segmentation is more difficult when features are not spatially registered. Figure 1b shows a multicolored cube. Which feature should be the most compelling, the color or the geometric lines indicating the cube? In the general case this answer depends on the goals of the perceiver. Another common problem occurs when an object is occluded (Figure 1c); a theory of low-level vision must be able to explain how an object is seen as a segment when the features are only partially registered or incomplete. Real image data is also noisy and many segments are only perceived owing to the combination of weak evidence of several features. The evidence may be so weak that each feature, if viewed in isolation, would be uninterpretable.

Figure 1

We develop the nucleus of a theory of low-level and intermediate-level vision which explains the above aspects of segmentation in terms of *massively-parallel* cooperative computation [Rosenfeld et al., 1976; Zucker, 1976; Marr, 1979] between two groups of networks. One group, *intrinsic images* [Barrow and Tenenbaum, 1978], can be computed primarily in terms of local constraints. The other, termed a *feature space*, can be computed primarily in terms of global mappings from intrinsic images to feature space. Feature space itself may have many different levels of abstraction. Intrinsic images and feature spaces are collectively called *parameter networks* because they both have a common organization. That is, the network is an organization of basic units, each representing values of a particular parameter. The simple structure of units simplifies the control task and also makes the network representation easily extendable. The basic elements of the theory are the following.

1) The cooperative computation of several intrinsic images.

Recent work has shown how to calculate intrinsic images from raw intensity data. Examples are images of velocity (optical flow) [Horn and Schunck, 1980; Ullman, 1977; 1979], surface orientation [Horn and Sjoberg, 1978; Ikeuchi, 1980], occluding contour [Prager, 1980; Rosenfeld et al., 1976], and disparity [Marr and Poggio, 1976; Barnard and Thompson, 1979]. Intrinsic images can be computed independently under special conditions, but in general they are interdependent. Intrinsic images are in concert with the hypothesis that the visual system builds many intermediate descriptions from image data. These descriptions represent important parameters such as velocity, depth, surface reflectance explicitly, since in the explicit form they are easier to map into object descriptions.

2) The extraction of useful parameters from intrinsic images.

If parts of the intrinsic image are organized in some way, this organization can be detected by a general Hough transform technique [Duda and Hart, 1972; Ballard, 1981a; Kender, 1978; Ohlander et al., 1979]. This is done by describing the organization in terms of parameters and then mapping the intrinsic image points into parameter space. The transformation will be many-to-one onto

parameter values which represent meaningful units. A major advantage of the Hough transform is that it is relatively insensitive to occlusion and noise.

3) Interactions involving several levels of abstraction.

The Hough transform is a way of seeing spatial information as a unit. However, if the unit has a complex structure the mapping from space to unit can be unmanageably complex. A way around this is to introduce units of intermediate levels of abstraction [Sabbah, 1981; Ballard and Sabbah, 1981; Kender, 1978]. This reduces a complex transform to several simpler transforms between units at successively higher levels of abstraction.

4) Focus-of-attention mechanisms.

Visual focus-of-attention can be partly explained as the conjunction of two mechanisms: 1) the use of Hough transforms to modify sensor input; and 2) the *sequential* application of Hough transforms.

5) Coupling between intrinsic images and parameters.

In general, intrinsic images *cannot be computed* without global parameters. At the same time, these global parameters are what we mean by seeing parts of the intrinsic image as a segment. In these cases the intrinsic image and parameters are said to be *tightly coupled*; although each cannot be computed independently, they can be computed simultaneously [Ballard, 1981b; 1981c].

We re-emphasize that our interest is low-level vision. Thus in item (4) above, focus of attention is interpreted in a narrow sense: visual features which are clear can help the recognition of other features (or perhaps direct eye movements). We do not attempt to explain general plans and goals.

Representations for Parameter Networks

The basic element of a parameter network is a **parameter node**. A parameter node will represent a single parameter *value* and has an associated confidence. The value is a set of numerical measurements for the node; the confidence is a measure of their believability. For example, if there is an edge at (10,10) with orientation 30° and length 5 units, the vector value of the parameter node representing the edge is $(x,y,O,s) = (10,10,30°,5)$. The associated confidence is a measure of the fuzziness of this estimate. One way a confidence may be increased is if there are nearby edges of the same orientation which align. Thus in Figure 2 the edges in (a) and (b) have the same value but we can be more confident in case (b).

Figure 2.

This paper assumes a very simple model; namely, collections of value units. Each value unit is connected to a subset of other value units, and can alter only those units. Underlying *physical principles* determine the appropriate connection subsets. The confidence updating is done by non-linear relaxation. The overall structure of the paper is slanted towards abstractions of physical principles; however, we also show how these principles are implemented in the networks.

2. Intrinsic Images

An intrinsic image is an image of some important parameter that is *in registration* with the original intensity image [Barrow and Tenenbaum, 1978; Marr, 1979], that is, each parameter is indexed by retinal coordinates. For example, in the velocity (optical flow) image, one is able to compute at each point in time and for each spatial position a local velocity vector $v(x,t)$. Figure 2 shows Horn's example for a rotating sphere [Horn and Schunck, 1980]. Intrinsic images may only be computable over certain parts of the image, and over those parts the parameters are *continuously*

varying. While intrinsic images are not segmented into parts of objects, they are distinctly easier to segment than the original intensity image. Other examples of such images are surface orientation, occluding contour, and disparity.

Figure 3.

Very recently there has been rapid progress in finding algorithms for computing intrinsic images from intensity data. What is remarkable is that each such image type is computed in the same manner. Two constraints, one derived from physical principals and the other from a constraint that the resultant images should be locally smooth, suffice to specify a parallel-iterative algorithm. Table 1 shows this commonality but is not an exhaustive list of approaches. See page 2 for additional references.

Table 1: Intrinsic Images

Parameter	Physical Constraint	Smoothness Constaint	Refs.
Edge Orientation θ	boundaries are locally linear	nearby edges should align	Prager 1979
Disparity d	if x corresponds to x' then $f(x+\Delta)=f(x'+\Delta)$	neighboring points should have similar disparities	Marr and Poggio 1976
Surface Orient'n. θ,φ	$f(x)=R(\theta,\varphi,\theta_S,\varphi_S)$ θ_S,φ_S is the light source direction	$\nabla^2\theta=0$ $\nabla^2\varphi=0$	Ikeuchi 1980
Optical Flow u,v	$df/dt=0$	$\nabla^2 u=0$ $\nabla^2 v=0$	Horn and Schunck 1980

While the above algorithms work well on images which are constrained to satisfy the underlying assumptions, they may not work in the general case. Almost always there are free parameters or boundary conditions which have to be determined independently.

2.1 Multi-Resolution Relaxation Methods

One general notion of "boundary condition" is image resolution. Previous methods for computing intrinsic images have used a single image resolution, but in most situations this is unrealistic. What is the correct resolution? At high resolution

* noise is a factor
* convergence is slow
* basic assumptions may not hold

To see the last point, imagine a surface with a micro-texture. At low resolution the surface structure is blurred and simple reflectance models hold, but at high resolution the microstructure can render such models useless. At low resolution

* noise is less of a factor
* convergence is fast
* basic assumptions may not hold

The last point arises from the fact that most intrinsic images are computed from constraints which assume local variations are smooth. With increasing grid resolutions, these assumptions are less likely to be valid.

Hence a conjecture is that there is a range of resolutions for which the computations will be valid. Furthermore, this range is expected to be spatially variant. A tool for exploring this conjecture is multigrid relaxation techniques [Brandt, 1977], which have proven very useful for solving differential equations. This model, together with reasoning from physical first principles, should allow the determination of image-dependent grid resolutions for which intrinsic image computations are valid. Multigrid techniques are of course related to pyramids [Tanimoto and Pavlidis, 1975; Hanson and Riseman, 1978].

2.2 Cooperative Computation of Multiple Intrinsic Images

Intrinsic images are logically computed simultaneously. In fact, they have to be; otherwise each intrinsic image is underdetermined in the general case. (Only on certain synthetic images is the computation well-defined.) Furthermore, they are highly interdependent, particularly at points of discontinuity [Barrow and Tenenbaum, 1978]. For example:

- intensity edges can be indicative of depth discontinuities. Thus the edge image is coupled to the disparity image;
- surface orientation is also indicative of depth discontinuity and is thus related to the other two; and
- different objects which are moving relative to each other produce discontinuities in the flow field.

By incorporating these couplings in the intrinsic image computations, one should find general cases where the computations will converge. A separate issue is the behavior of the coupled computations in the face of conflicting information.

2.3 Intrinsic Images at Different Levels of Abstraction

The survey of intrinsic images (Table 1) excluded the fact that intrinsic images may have fine structure involving several levels of abstraction. In fact, it seems likely that multiple abstraction levels are necessary in many cases. For example, Zucker [1980] uses two levels of abstraction in computing orientation intrinsic images, one for points of high gradients and the other for edge segments. The computation of a velocity image in 3-d could involve three levels of abstraction:

- a *change detection* level where units are used for variations in intensity over space and time $\Delta I/\Delta x'$, $\Delta I/\Delta y'$, $\Delta I/\Delta t$ (primes denote retinal coordinates);

- an *optical flow* level where units correspond to retinal velocities $(u(x',y'),v(x',y'))$;

- a *3-d flow* level where units correspond to 3-d velocities $(v_x(x,y,z),v_y(x,y,z),v_z(x,y,z))$.

The feasibility of computing the optical flow from change measures has been studied by [Barnard and Thompson, 1979; Prager, 1980; Horn and Schunck, 1980]. The feasibility of computing 3-d flow is explored in [Ballard, 1981c].

2.4 Intrinsic Images and Parameter Nodes

Two models have been used to compute intrinsic images: 1) the value unit defined in Section 1 [Prager, 1980; Marr and Poggio, 1976]; and 2) a variable unit [Ikeuchi, 1980; Horn and Schunck, 1980]. In the first model there is a unit for every value of every variable; in effect the representation has only constants. Constant value units may have outputs which are confidences between zero and one. In the second model, each unit represents a variable which can take on values (the standard method is to use an array for these units). The output is the value; there is no explicit notion of confidence.

In general the unit/value representation is sufficient since problems formulated to use variables can be transformed into unit/value problems in the following manner. Suppose x, y, and z satisfy a relation $R(x,y,z) = 0$. Let us use a set of values A for x, B for y, and C for z. Where a $\in$ A, we would like C(a) to be 1 if there exist b $\in$ B and c $\in$ C such that $C(b) = 1$, $C(c) = 1$, and $R(a,b,c) = 0$. To implement this in a parameter network connect all pairs of (b,c) $\in$ BxC to a value (a) if $R(a,b,c) = 0$. Then starting with initial confidences, increment C(a) if there exist (b,c) such that $R(a,b,c) = 0$ and $C(b) + C(c) >$ some threshold. The individual values b and c may be treated similarly.

Note that the updating function is *nonlinear*, when the underlying physical relation R is nonlinear. If the relation R can be linearized then the cooperative computations can be shown to be equivalent to linear programming [Hinton, 1979]. The linear case has also been analyzed by [Hummel and Zucker, 1980].

3. Parameter Spaces

What does it mean to perceive parts of an image as a segment? In our theory, this perception takes place if there is a parameter space such that each of the parts can have the same parameter value. This general idea is illustrated by the following examples.

* Parts of a *color* image may be seen as a segment if they have the same color. In this case the parameter space is a space of colors and the parts map into a common point representing the common color.

* Parts of an *optical flow* image may be seen as a segment if they are part of a rigid body that is moving. In this case the parameter space represents the rigid body motion parameters of translational and rotational velocity and parts of the image map into a common point in that space.

* Parts of *edge* and *surface orientation* images may be seen as a segment if they are part of the same shape. This case is more complicated as there must exist some internal representation of the shape. Given this representation, the parameter space represents the transformation (scale, rotation, translation) from the internal representation to the (viewer-centered) image representation. Parts of the image which are seen as the shape have common values for these parameters.

A general way of describing this relationship between parts of an image and the associated parameters is the *Hough transform* [Hough, 1962; Duda and Hart, 1972; Kimme et al., 1975; Shapiro, 1978]. In our low-level vision theory, Hough transforms relate intrinsic images and feature spaces and feature spaces at different levels of abstraction. If the intrinsic image parameter is a vector $(x,a(x)) \in$ A and an element of feature space is a vector b $\in$ B then there is usually a *physical constraint* that relates a(x) and b, i.e., some relation f(a,b) such that

$$f(a,b) = 0.$$

The space A represents all possible intrinsic image values. A particular intrinsic image is described by a set of values $\{a_k\}$ where $a_k = a(x_k)$. Now the set $\{a_x\}$ is only consistent with certain elements in the space B, owing to the constraint imposed by the relation f. This physical constraint can be exploited in the following manner. For each a_k we can compute the set

$$B_k = \{b \mid a_k \text{ and } f(a_k,b) \leq \delta_b\}$$

Define H(b) as the number of times the value b occurs in $U_k B_k$. H(b) is the Hough transform from the space a to the space b and is the number of points in intrinsic image space which are consistent with the parameter value b. H(b) makes the most sense when the values both (a(x),x) and b are discrete. Hence the constant δ_b above is related to the quantization in the space B. H is also

best normalized by defining $C(b) := H(b)/\Sigma_b H(b)$. In that case, the value $C(b)$ can stand for the confidence that the segment with feature value b is present in the image.

Concerning the implementation of Hough transforms in networks, $B_k \subset B$ is the subset of B units to which the unit a_k hould be connected in the network. A separate Hmax unit is needed for normalization.

The Hough transform need not originate from intrinsic image space but can be defined between any two spaces A and B as long as there is some relation $f(a,b) = 0$ for $a \in A$ and $b \in B$. To avoid describing the above computations in detail, we can use a shorthand notation for Hough transforms. Each transform can be described as the triple

$\langle a,b,f \rangle$

where the necessary computations are implicit. Note that the order of a and b is important in the notation; in general, $\langle a,b,f \rangle$ is not equivalent to $\langle b,a,f \rangle$.

As a very simple example of a Hough transform, we describe how a patch of red in an image may be seen as a unit. For this to happen, an association is made between the spatially contiguous points in the image and the particular value "red" in a parameter space of colors. There are essentially three dimensions to color space. Although r-g-b is widely used in computer applications, humans seem to use an opponents-process basis (r-g, y-b, white-black) [Hurvich and Jameson, 1957]. One (admittedly overly simplified) way of transforming from (r,g,b) space to opponents color space is to use the following linear transformation:

$$
\begin{bmatrix} rg \\ yb \\ bw \end{bmatrix} = \begin{bmatrix} 1 & -2 & 1 \\ -1 & -1 & 2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix} \tag{3.1}
$$

Thus the Hough transform is given by $\langle a,b,f \rangle$ where

$a = (r(x,y), b(x,y), g(x,y))$

$b = (rg, yb, bw)$

and

$f = Ta - b$

where T is the matrix defined by Eq. 3.1.

For a red spot on a green background there are two values of color parameters which have high values for $C(b)$: red and green. The rest of color Hough transform has low values. Figure 4 shows this idea, which has been used by [Hanson and Riseman, 1978; Ohlander et al., 1979], applied to a color image.

Figure 4: Hanson and Riseman's
Segmentation in Color Space.

To show that intrinsic images and parameter spaces may be related in more complicated ways, we briefly describe an example of how a specific two-dimensional shape is detected by specifying a Hough transformation from edge space (local linear edges detected with a standard edge detector) to a four-dimensional parameter space consisting of local origin coordinates, rotation and scale. Both the color-space example and this one have the same solution at an abstract level. In each case there is a transformation from intrinsic image space to parameter space that segments the image. In the first case, points in the color image have the same color values. In the second case, points in the edge image have the same shape parameter values. In fact, almost all segmentation problems can be characterized in this fashion.

Table 2 shows some other Hough transforms.

Table 2: Hough Transforms

Intrinsic Image	Hough Transform
Optical Flow	* Heading * Rotation of 3d rigid body
Surface Orientation	* Illumination angle * Shape
Occluding Contour	* Shape * Surface orientation
Disparity	* Segments of constant disparity
Color	* Segments of constant color

The two-dimensional shape example shows the general feature of Hough transforms: if the algorithms are completely parallel, the space required is exponential in the number of parameters. This can lead to immense space requirements. For example, consider an eight-parameter space of 100 discrete values for each parameter. The total number of parameter nodes required to represent the space is 100^8! Fortunately this problem can generally be alleviated by detecting groups of parameters sequentially. The example of 2-d shape detection is reconsidered in Section 3.2 to illustrate this extremely powerful decomposition technique.

3.1 Detecting Two-Dimensional Shapes

Two-dimensional shapes can be found from a *primal sketch* [Marr, 1978] by encoding the shape information in *constraint tables* [Ballard, 1981a]. Consider the case where an object being sought has no simple analytic form, but has a particular silhouette. Suppose for the moment that the object appears in the image with known shape, orientation, and scale. (If orientation and scale are unknown, they can be handled as additional parameters, as we will show.) Now pick a coordinate system for the silhouette and draw a line to the boundary from the coordinate system origin. At the boundary point we can compute the gradient direction and length and store the reference point as a function of this information. Thus it is possible to precompute the location of the reference point from boundary points given the gradient angle. The basic strategy of the Hough technique for shapes is to compute the loci of points in parameter space from an edge in image space and increment those points in an array. Figure 5 shows the relevant geometry.

Figure 5: Geometry for the Hough Transform.

In this case the reference point coordinates (xc,yc) are the only parameters (remember, rotation and scaling have been fixed). Thus if we encounter in an image an edge point (x,y) with gradient orientation (φ) and span (!) we know that the possible reference points are at

$$(x + r(\varphi,l)\cos(\alpha(\varphi,l)), y + r(\varphi,l)\sin(\alpha(\varphi,l)))$$

and so on.

Thus we can describe the generalized Hough algorithm as follows:

Generalized Hough for Shapes

Step 0. Make a table for the shape to be located like that shown in Figure 2.

Step 1. Form an array of possible reference points

$H(xc_{min}:xc_{max}, yc_{min}:yc_{max})$ initialized to zero.

Step 2. For each edge do the following:

Step 2.1. Compute $\varphi(x), l(x)$

Step 2.2a. Calculate the possible centers, i.e., foreach table entry for (φ, l) compute

$$xc := x + r(\varphi, l)\cos(\alpha(\varphi, l))$$

$$yc := y + r(\varphi, l)\sin(\alpha(\varphi, l))$$

Step 2.2b. Increment the array

$$H(xc, yc) := H(xc, yc) + 1$$

Step 3. Possible locations for the shape are given by maxima in the array H.

In terms of our Hough transform notation, the transform is of the form

$$\langle (\varphi(x,y), l(x,y), x, y), (xc, yc), T \rangle$$

where T is the constraint relation between $(\varphi(x,y), l(x,y), x, y)$ and (xc, yc) shown by Figure 5. Also the inner loop of the algorithm (Step 2.2) computes B_K given an edge (φ^K, l^K). The outer loop (Step 2) computes $U_K B_K$. The results of using this transform to detect a shape are shown in Figure 6. Figure 6a shows an image of shapes. The R-table has been made for the middle shape. Figure 6b shows the Hough Transform for the shape, i.e., $H(xc, yc)$ displayed as an image. Figure 6c shows the shape given by the maxima of $H(xc, yc)$ overlaid on top of the image.

Figure 6: Applying the Generalized Hough Technique.
(a) Synthetic image. (b) Hough Transform
$A(xc, yc)$ for middle shape.

What about the parameters of scale and rotation, s and θ? These are readily accommodated by expanding the accumulator array and doing more work in the incrementation step. Thus, in Step 1, the accumulator array is changed to

$$H(xc_{min}:xc_{max}, yc_{min}:yc_{max}, s_{min}:s_{max}, \theta_{min}:\theta_{max})$$

and Step 2.2a is changed to

```
foreach table entry for (φ,l) do
    foreach s and θ
    xc := x + r(φ,l) s cos(α(φ,l) + θ)
    yc := y + r(φ,l) s sin(α(φ,l) + θ)
```

Finally, Step 2.2b is now

$$H(xc, yc, s, \theta) := H(xc, yc, s, \theta) + 1$$

Now the transform is given by $\langle (\varphi(x,y), l(x,y), x, y), (xc, yc, s, \varphi), T' \rangle$ where T' incorporates the rules for computing s and φ. Notice that this algorithm is notionally parallel since all the incrementations are independent, and that the space required is exponential in the number of parameters.

3.2 Feature Space Decompositions

In the example of Section 3.1, a particular shape is found by a notionally parallel transform from edge space $(\varphi(x,y), l(x,y), x, y)$ to a four-dimensional shape space (xc, yc, s, θ). However, time can be traded for space by finding *groups* of these parameters *sequentially*. The advantage of the sequential search is that the dimensionality of the computation at each stage is much less than the

single computation involving all of the parameters simultaneously [Ballard and Sabbah, 1981]. For example, where N_x, N_s, and N_θ are the sizes of the spaces (xc,yc), s, and θ respectively, searching for a particular shape's parameters in the order (s,θ) and (xc,yc) requires parameter space equal to $N_s N_\theta + N_x$ instead of $N_s N_\theta N_x$. The Hough transform for the individual group is still notionally parallel, so the time needed in the sequential transform is only proportional to the number of parameter groups. In the shape example, the number of groups is two.

To see how scale and orientation can be detected independently, consider a table that encodes the orientation of the edge with respect to the silhouette's coordinate system. For example, for each edge (φ,l) encode the angle necessary to rotate the edge clockwise so that it is parallel to the x-axis. Using this table, the algorithm is as follows:

Hough Algorithm for Orientation and Scale:

Step 0. Make an orientation table as a function of φ and l.

Step 1. Form an array of possible scale-orientation pairs $H(0:2\pi, S_{min}:S_{max})$.

Step 2. For each edge do the following:

Step 2.1 Compute $\varphi(x), l(x)$.

Step 2.2 Foreach S do the following:

(a) look up the table entry $\alpha(\varphi(x), s^* l(x))$.

(b) increment the array

$H(\alpha,s) := H(\alpha,s) + 1$.

Step 3. Possible orientations and scales are given by maxima in the array H.

The value of sequential searches through parameter space becomes even more important in 3-d since this case requires seven parameters: three positional coordinates; three orientation angles; and a scale factor. The sequential Hough-shape transform extends readily to 3-d and has been used to detect polyhedra [Ballard and Sabbah, 1981] using the constraints of [Kanade, 1978; 1979].

The previous example is for a single shape. For N shapes, given that the search is in parallel, a size factor of N is added to the search space. To cut down on the impact of this factor one needs a shape taxonomy like that of Bribiesca [Bribiesca and Guzman, 1979] where all shapes can be described as a branch in a single shape tree. The advantage of the shape tree is that rather than looking for all N shapes in parallel, the search can be partitioned into searches of spaces of size N_i, N_{ij}, N_{ijk}, etc., where the sum of these is roughly equivalent to $\log(N)$.

4. Hierarchies of Abstraction Levels

The value of using several hierarchical levels of abstraction in vision is that the interaction between levels is simplified. This does not mean that high-level descriptions cannot influence low-level descriptions, or that the entire computations are not carried out in parallel. Rather, each descriptive level can only influence nearby levels. In Sabbah [1981], the limitation is to levels directly above and below. Other levels are influenced indirectly. The implication for the Hough transforms, which specify the constraints between levels, is that the constraint relationships between levels involve only a few parameters. This is an especially important feature, since the space required by the Hough transform is exponential in the number of parameters, as are the sets $\{B_K\}$. Different levels of abstraction have been used by [Hanson and Riseman, 1978]. Examples using the Hough transform may be found in [Sabbah, 1981; Kender, 1978]. Sabbah uses four levels such as those shown in Figure 7 to reorganize origami world figures.

Figure 7.

To show an example in detail, Kender's technique for detecting vanishing points in an images from oriented line segments [Kender, 1978] is described. Such line segments which are part of a given vanishing point form a radial field which emanates from the point. Different vanishing points have different sets of associated radial line segments (Fig. 8).

Figure 8.

This example is interesting since the same situation occurs with respect to optical flow due to pure translation. If the objects in the image are stationary with respect to a translating observer, then the flow vectors will be emanating radially from a "focus-of-expansion" (FOE) in the direction of motion. Objects translating with respect to the observer's frame will produce their own flow emanating from a different FOE (Fig. 8).

This example involves two levels of abstraction. The first transforms colinear edge segments into points (representing lines). Radial sets of edge elements correspond to circles through the origin in line-space. Thus the second transformation is between circles in line-space to points in radial-field space.

The first level is easy if a (ρ,θ) line space is used where

$$\rho = x \cos\theta + y \sin\theta.$$

Since an edge element has direction α (Fig 9), each such element maps onto precisely one point in (ρ,θ) space: $(x \cos\alpha + y \sin\alpha, \alpha)$. Thus the Hough transform, in the notation of Section 3, is:

$$\langle(x,y,\alpha(x,y)), (\rho,\theta), (\theta = \alpha; \rho = x \cos\alpha + y \sin\alpha)\rangle.$$

Figure 9.

Now maxima in $C(\rho,\theta)$ correspond to lines in the image. Also, radial lines will form a circle of local maxima in (ρ,θ)-space. To see this note that the triangle OPQ in Figure 1b is always a right triangle, and therefore OQ must be the diameter of a circle. Note that this circle is constrained to go through the origin so that its diameter must be on the line

$$\rho/2 = a \cos\theta + b \sin\theta$$

where $(2a,2b)$ is the location of the focus of expansion (or vanishing point). Thus the second transform is

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

Implementation in Parameter Networks

In our earlier definition of the Hough transform, we assumed that the measurements a_k all had confidence equal to unity. With higher level of abstraction Hough transforms, this may no longer be the case. This is easily handled by keeping track of the confidences in the set B_k, i.e.,

$$B_k = \{(b,C) | f(a_k,b) \leq \delta_B \text{ and } C = C(a_k)\}.$$

Then $H(b)$ is the sum of the confidences associated with the value b in $U_k B_k$.

5. Focus-of-Attention

Previously, intrinsic image to feature space transforms used single Hough transforms. We are now ready to tackle issues which arise when several Hough transforms are used. First we show that multiple Hough transforms can be invoked in parallel to resolve the problem of detecting a unit with multiple features. This is done via the mechanism of a *context* Hough transform which is the sum of individual Hough transforms. Next, we describe a focusing mechanism which exploits the fact that an ambiguity in one space may be resolved in another. This technique allows the detection of arbitrarily fine detail. Attention can be directed from a unit to its subparts and back again via a mechanism termed *sequencing*.

5.1 Spatial Context

If a unit has multiple spatially registered features, these can be detected by applying two different sets of Hough Transforms. The Hough Transform defined in Section 3 is bottom-up: points in the intrinsic image space determine plausible sets of points in feature space. The complementary transform is top-down: points in feature space determine plausible sets of points in intrinsic image space. Formally, given a set $\{b_k\} \in B$, we compute

$$A_k = \{a \mid b_k \text{ and } f(a,b_k) \leq \delta_A\}$$

$H(x)$ is the number of times the value $a(x)$ occurs in $U_k A_k$. The *mapping* which defines $H(a)$ is likely to be one to many and furthermore, for a given feature, different b_ks should give rise to disjoint subsets of A. Owing to this last point, it is intuitively appealing to deal with $H_a(x)$ which is simply the sum of the confidences of different values of the parameters $a_1, a_2,...$ which are at the same spatial location x,y, i.e.,

$$H_a(x) = \Sigma a \ H(a,x)$$

An Example

Consider again the image of a red spot on a green background, where the spot takes up one-third of the image pixels. Then the transform $H(b)$ where $b = r,g,b$ has two peaks and is zero everywhere else, i.e., for four-bit color scale accuracy

$$H(b) = \begin{array}{ll} 1 & \text{if } b = (0,15,0) \\ 1/2 & \text{if } b = (15,0,0) \\ 0 & \text{otherwise} \end{array}$$

Now consider $b_1 = (15,0,0)$ and compute $H(a,x)$. This is given by

$$H(a,x) = \begin{array}{ll} 1 & \text{if } x \text{ in spot and } a = RED \\ 0 & \text{otherwise} \end{array}$$

A point in A represents the single color red and so $H(x)$ in this case is

$$H(x) = \begin{array}{ll} 1 & \text{if } x \text{ is spot} \\ 0 & \text{otherwise} \end{array}$$

The transform $H(x)$ is called the spatial context transform for reasons that will become more apparent when we discuss focus of attention. The *effect* of this transform is to place an imaginary filter in front of the sensors. In the above case, only sensors that are spatially registered with RED sensors would receive input.

Multiple Features

Now consider the case where multiple features are present in the image. Each individual bottom-up transforms for different features can be applied in parallel to compute $H_1(b_1)$, $H_2(b_2),...,H_m(b_m)$ (for m feature spaces). Now maxima in each of these spaces can be used to compute individual top-down transforms $H_{a1}(x),H_{a2}(x),...,H_{am}(x)$. The generalized spatial-context transform $H(x)$ is simply the normalized sum of these individual transforms, i.e.,

$$H(x) = (1/m) \ \Sigma_k \ H_{ak}(x)$$

Now the value of $H(x)$ at a point x is the *fraction of the maximum number of spatially registered features that are present*. High values of H correspond to spatially registered intrinsic image points which each have been grouped into a unit by a separate bottom-up transform. Thus $H(x)$ represents a possible solution to the multiple-feature problem.

5.2 Subspaces and Sequencing

In our formalism, a segment in an image is ideally represented as a conjunction of Hough transform maxima. Each set of maxima corresponds to an organization with respect to a given modality: color, velocity, etc. In the previous section we showed how the parallel generation of these maxima could be used to discover regions in the image corresponding to multi-modal units. Unfortunately, this technique will usually be inadequate because the unit is not manifested as a clear maxima in all the modalities. As an example, consider a light-blue, moving unit, against a background of other units, none of which are light-blue, but which are moving. In the color space, the unit is clearly revealed; light-blue units have high confidence values (Fig 10).

Figure 10.

In velocity space, however, there is no clear maximum owing to the presence of other moving units.

The fundamental problem is that each modality consists of a *projection* of feature space. In the high-dimensional space consisting of the concatenation of all the individual dimensions of each modality, each unit would appear as a distinct maximum. The visual system model is structured to examine only the subspaces of the individual modalities. The principal reason for this is economy; the space requirement increases exponentially with the number of modalities.

This problem can be surmounted if the different parameter spaces are examined sequentially. First the parameter spaces are examined for maxima. The most distinct maxima is picked and its inverse Hough transform, $C(x)$, is generated. This transform can be usd to block input from sensors positioned at its low confidence values. To see how this might work, let us reconsider the previous example of the light-blue, moving unit. In color space there is a clear maximum corresponding to light-blue. This value is used to generate $C_{light-blue}(x)$ and block input from all sensors that are not spatially registered with light-blue color input. The net effect is that in velocity space there is now a clear maxima as input from other units has been blocked.

5.3 Multiple, Spatially-Registered Features

Sequencing solves the problem of building up coherent groups of features, but has its drawbacks. For example, if the "blue," "moving," "horizontal" object were a "frisbee," one would like this percept to be triggered via a Hough-like transform. However, in the sequencing example, there is initial evidence for all light-blue objects, and this is a very large set. Worse, the percept "frisbee" could be triggered by non-spatially registered groups of "blue" and "moving" inputs.

There is a solution to these problems if we assume that, in general, actual occurrences of features will be sparse. In other words, in a given image there should not be two very similar colors associated with different objects. If there are, our Hough transform model will only be able to concentrate on one of them at a time.

The solution is due to [Feldman and Ballard, 1981]. We will develop it here in terms of our Hough transform formalism in stages. First we formally concatenate parameter spaces. Next we describe *low-resolution* concatenated spaces. Finally, we consider low-resolution parameter spaces which can be *tuned* to specific parameter values.

Ideally, one could resolve the spatial registration problem by concatenating feature spaces. For example, concatenating color space with motion space leads to

$$B_k = \{(b_c,b_m) | a_{kc}(x_k), a_{km}(x_k), f_c(a_{kc},b_c)\leq\delta_c, f_m(a_{kc},b_m)\leq\delta_m\}$$

where elements in the expanded space $(b_c,b_m) \in B_c \times B_m$ are only included if the input features are spatially registered, i.e., while this is simply described in symbolic form, it is also impractical since the parameter spaces for the combined-modality elements are impractically large. A partial solution to the size problem is to decrease the number of parameter nodes. Let b_c', b_m' be values for color and motion parameters respectively in the low-resolution spaces. Then the low resolution Hough transform is given by

$$B_k = \{(b_c',b_m') | a_{kc}(x_k), a_{km}(x_k), f_c(a_{kc},b_c')\leq\Delta_c, f(a_{km},b_m')\leq\Delta_m\} \quad (5.1)$$

where the bounds Δ_c and Δ_m are larger to account for the lower-resolution in parameter space. The grain of the low-resolution space can always be chosen to make the transform practical in terms of space. However, now groups of parameters that are sufficiently similar may be transformed into the same parameter node via Eq. (5.1). To resolve this problem we use a two-tiered transform, consisting of high-resolution single-modality transforms and low-resolution multi-modality transforms. Using the single-modality transforms, we select maxima $\{b_c^*\}$ and $\{b_m^*\}$ such that

$$b_c^* = \max_{bc}\{b_c \in b_c' \pm .5\Delta_c\}$$

and

$$b_m^* = \max_{bm}\{b_m \in b_m' \pm .5\Delta_m\}.$$

These values are then used to *tune* the low-resolution Hough transform, i.e.,

$$B_k = \{(b_c',b_m') | a_{kc}, a_{km}, f_c(a_k,b_c^*)\leq\Delta_c, f_m(a_k,b_m^*)\leq\Delta_c'$$

Thus the low resolution transform can be tuned to count only a subset of the high resolution parameter nodes. The drawback of this technique is that it can only respond to a single value of (b_c,b_m) in each range $\{b_c \pm .5\Delta_c, b_m \pm .5\Delta_m\}$. Thus either the high confidence parameter nodes must be sufficiently sparse, or only one of the confusion classes can be examined at any one time. This disadvantage is outweighed by being able to detect spatially-registered features and thus circumvent the more severe problem discussed earlier.

6. Tight Coupling

Most of the previous examples imply that the various Hough transforms are relatively independent. That is, once the intrinsic images are computed, the transforms can be computed. The general case is that this is not true; the intrinsic image contains global parameters which must be computed using Hough transforms. Since the Hough transform required an intrinsic image it might seem that neither could be computed. In fact, both the Hough transform and the intrinsic images can be computed by incorporating the Hough transforms into the parallel-iterative scheme used to compute the intrinsic images. If the combined problem is well-conditioned: 1) the partial result for the intrinsic image will be sufficient to produce a partial result for the Hough transform, and vice versa; and 2) this process of using partial results in a parallel-iterative manner will converge. We term this interdependence tight coupling and illustrate it with two examples.

In the first example, we show how a surface orientation intrinsic image can be computed from intensity information. This example seems paradoxical at first since to compute surface orientation one must know the location of the source of illumination and vice versa. We show how both these computations can be conducted simultaneously with the partial result for the surface orientation helping the illumination angle determination, and the partial result for the illumination angle helping the surface orientation determination. The illumination angle is determined by a Hough transform.

In the second example, we show that a three-dimensional flow field can be segmented into groups of vectors that represent general rigid body motion. The problem here is that an individual field vector $v(x)$ is an unknown sum of rotational and translational components, i.e., $v(x) = v_R(x) + v_T(x)$. These components can only be determined by knowing global rigid body motion parameters. However, these parameters can be determined only if $v(x)$ is partitioned into $v_R(x)$ and $v_T(x)$. As in the earlier example, this problem can be resolved by a parallel-iterative scheme which computes both the global parameters and the velocity-field decomposition simultaneously.

Rather than being isolated examples, tight coupling is believed to be the general case. Extending the scope of the parallel-iterative computation is the general solution.

6.1 Shape from Shading by Relaxation

Given the orientation of a surface with respect to a viewer, its reflectance properties and the location of a single light source, that the brightness at a point of the viewer's retina can be determined. That is, the reflectance function $R(\theta,\varphi,\theta_s,\varphi_s)$, where θ,φ and θ_s,φ_s are orientations of the surface and source respectively, allows us to determine $I(x,y)$, the intensity in terms of retinal coordinates [Horn and Sjoberg, 1978]. The form of R is assumed to be known. However, the perceptual problem is the reverse: given $I(x,y)$ and $R(...)$, determine $\theta(x,y),\varphi(x,y)$ and θ_s,φ_s.

In general, the problem of deriving $\theta(x,y)\varphi(x,y)$ and $\theta_s\varphi_s$ is underdetermined. However, Ikeuchi [1980] showed that the surface could be determined locally once $\theta_s\varphi_s$ was specified. This method has been extended [Ballard, 1981b] to the case where $\theta_s\varphi_s$ is initially unknown.

The algorithm is outlined as follows. For a single light source, the intensity at a point on a retina can be described in terms of the orientation of the normal of the corresponding surface point and the surface orientation. That is, in spherical notation,

$$I(x,y) = R(\theta,\varphi,\theta_s,\varphi_s)$$

where the angles θ and φ are functions of x and y. Now by minimizing $(I-R)^2$ and appending a smoothness constraint on θ and φ we have [Ikeuchi, 1980] an expression for the *local* error (if the estimate for θ and φ is unreliable) as follows:

$$E(x,y) = (I-R)^2 + \lambda((\nabla^2\theta)^2+(\nabla^2\varphi)^2)$$

where λ is a Lagrange multiplier. For a minimum, E_θ and $E_\varphi = 0$. Skipping some steps, this leads to

$$\varphi(x,y) = \varphi_{ave}(x,y)+T(x,y)R_\varphi$$
$$\theta(x,y) = \theta_{ave}(x,y)+T(x,y)R_\theta$$

where $\varphi_{ave}(x,y)$ is a local average and

$$T(x,y) = (1/16\lambda)(I-R)$$

In solving these equations, we assume θ_s and φ_s are known. An iterative method is used where the φ_{ave} and θ_{ave} are calculated from a previous iteration.

To calculate θ_s and φ_s, we assume θ and φ are known and use a Hough technique. First we form an array $H[\theta_s,\varphi_s]$ of possible values of θ_s and φ_s initialized to zero. Now we can solve the reflectance equation for φ_s. The Hough technique works as follows. For each surface element θ,φ, and for each θ_s we calculate φ_s and increment $H[\theta_s,\varphi_s]$, i.e., $H[\theta_s,\varphi_s] := H[\theta_s,\varphi_s]+1$. After all surface elements have been processed, the maximum value of C corresponds to the location of the point source. In [Ballard, 1981b] it is shown that calculation of the source location can proceed in parallel with that of $\theta(x,y)$ and $\varphi(x,y)$ and that the two calculations will converge.

Results for the one-dimensional case are shown in Figure 11 for the case of a small surface "bubble." Figure 11 shows the surface convergence, as well as the convergence of the illumination angle.

Figure 11: (a) Shading (top left curve).
(b) Surface convergence (colored points immediately below (a)).
(c) Illumination angle Hough transform (bottom left).
(d) Illumination angle convergence (upper right).

It is important to remember that the boundary conditions in this problem have been provided *a priori*; in this case they are the orientation of the surface at the boundary of the bubble. Generally, these will have to be determined by multiple intrinsic images relaxations, as mentioned in Section 2.

6.2 3-D Rigid Body Motion

The general motion of a rigid body can be described by eight parameters: three for translational velocity v_T; three for angular velocity Ω; and two for the location of the axis of rotation r. We describe the detection of rigid body motion in three parts, each of which uses Hough transforms. First, we show how to detect pure translation (v_T). Next we show how to detect pure rotation (Ω,r). Finally, we show that a 3-d flow vector can be iteratively decomposed into a translational component and a rotational component. These components are described by the parameters (v_T, Ω, r).

Pure Translational Motion

This case is very simple. If a rigid body is translating with velocity v_T, then a point on the body at location x will have velocity $v(x) = v_T$. To detect this take the Hough transform given by $\langle (x, v(x)), (v_T), (v(x)-v_T=0) \rangle$. The maximum value in $H(v_T)$ will correspond to the translational velocity.

Pure Rotational Motion

In the case of pure rigid-body motion, each point on an axis in space such that

$$v(x) = \Omega x \rho(x).$$

$$(6.1)$$

where v, Ω, and r are all orthogonal and $\rho(x)$ is a vector from the point x to the axis of rotation such that

$$\rho \cdot (\Omega x v) = 0.$$

That is, ρ is defined so as to be perpendicular to Ω and v.

One problem is to specify the axis of rotation. This is done using a vector r which is the smallest vector from the origin to the rotation axis (see Figure 12).

Figure 12.

The pure rotation case involves five parameters: three for the vector Ω and two to specify the axis of rotation. A standard Hough technique would involve a transformation from $(x, v(x))$ to (Ω, r') using Eq. (6.1). Only a vector r' equal to any two components of r is necessary since $\Omega \cdot r = 0$. However, a five-dimensional space is large, thus we are motivated to decompose the parameter space (Ω, r) into two, smaller spaces [Ballard and Sabbah, 1981]. One space is composed of two components ω_x and ω_y of a unit vector ω which defines the direction of Ω. The other is composed of the magnitude of Ω and two components of r.

Since ω must be perpendicular to v,

$$\omega \cdot v = 0.$$

Furthermore, $|\omega| = 1$. Combining these two equations leads to

$$\omega_x v_x + \omega_y v_y + \sqrt{(1 - \omega_x^2 - \omega_y^2)}v_z = 0, \tag{6.2}$$

which is a quadratic equation in unknowns ω_x and ω_y. Thus the direction of the rotation vector may be found from the Hough transform

$$\langle (v(x)), (\omega_x, \omega_y), (Eq. (6.2))\rangle.$$

Once ω is known, it can be used in the following series of equations. If $|\Omega|$ is the magnitude of the rotation vector, the vector s given by

$$s = x - \omega x v/|\Omega|$$

is on the rotation axis. Furthermore, r is given by

$$r = s - (s \cdot \omega)\omega$$

so that

$$r = (x - \omega x v/|\Omega|) - (x \cdot \omega)\omega. \tag{6.3}$$

This equation can be used to determine the first two components of r given a value for $|\Omega|$. Thus we can determine $|\Omega|$ and v from the following Hough transform:

$$\langle (x, v(x), \omega)\ (r_x, r_y, |\Omega|)\ (Eq. (6.3))\rangle.$$

General Rigid Body Motion

Finally, suppose the motion is completely general so that

$$v(x) = v_T(x) + \Omega x \rho(x).$$

Since only $v(x)$ can be measured, how can one determine how much is translational velocity and how much is rotational velocity? One possibility is to dynamically partition the velocity into two components $v_T(x)$ and $v_\Omega(x)$ which give the most consistent global parameters $(v_T, \Omega$ and $r)$. This would work as follows:

Step 0. Assume $v_\Omega(x) = v(x)$.

Step 1. Use the Hough transforms to estimate (Ω, r).

Step 2. Use (Ω, r) to determine $v_\Omega(x)$.

Step 3. Compute $v_T(x) = v(x) - v_\Omega(x)$.

Step 4. Use the Hough transform to estimate v_T.

Step 5. Compute $v_\Omega(x) = v(x) - v_T(x)$.

Step 6. If $v_\Omega(x)$ has not converged, go to Step 1.

7. Discussion

The key ideas of this paper are summarized in the introduction. Here we mention other ideas which do not fit easily under any one of the previous headings.

1) The Intrinsic Image/Feature Space Duality. By distinguishing between image fields and image features we know when relaxation is the more important tool and when the Hough transform is more important.

2) Unit/value. By reducing the underlying primitives to units of extreme simplicity we can algorithmically determine the connection patterns to represent m-ary relations.

3) Massive Parallelism. By assuming the availability of massive parallel computation, we reduce the need for sequential processing to more essential cases. For example, we use sequential processing in Section 5 to resolve real ambiguities in the input.

4) Extensibility. The representation is very general, being m-ary consistency relations, and can be extended to other domains besides vision, to arbitrary levels of abstraction within vision.

References

Ballard, D.H., "Generalizing the Hough transform to detect arbitrary shapes," TR55, Computer Science Dept, U. Rochester, October 1979; *Pattern Recognition*, 1981a.

Ballard, D.H., "Shape and illumination angle from shading," Computer Science Dept, U. Rochester, 1981b.

Ballard, D.H., "3-d rigid body motion from optical flow," Computer Science Dept, U. Rochester, 1981c.

Ballard, D.H. and C.M. Brown. *Computer Vision.* Prentice Hall, in press.

Ballard, D.H. and D. Sabbah, "On Shapes," submitted to 7th IJCAI, 1981.

Barnard, S.T. and W.B. Thompson, "Disparity analysis of images," TR 79-1, Computer Science Dept, U. Minnesota, January 1979.

Barrow, H.G. and J.M. Tenenbaum, "Recovering intrinsic scene characteristics from images," TN 157, AI Center, SRI Int'l, April 1978.

Brandt, A., "Multi-level adaptive solutions to boundary-value problems," *Math of Comp 31*, 138, 333-390, April 1977.

Bribiesca, E. and A. Guzman, "How to describe pure form and how to measure differences in shapes using shape numbers," *Proc.*, IEEE Computer Society Conf on Pattern Recognition and Image Processing, 427-436, Chicago, IL, August 1979.

Duda, R.O. and P.E. Hart, "Use of the Hough transform to detect lines and curves in pictures," *CACM 15*, 1, 11-15, January 1972.

Feldman, J.A., "A connectionist model of visual memory," in G.E. Hinton and J.A. Anderson (Eds), *Modeling Psychological Processes with Multiple Simple Units.* Hillsdale, NJ: Lawrence Erlbaum Associates, 1981.

Feldman, J.A. and D.H. Ballard, "Computing with connections," TR72, Computer Science Dept, U. Rochester, 1981.

Feldman, J.A. and Y. Yakimovsky, "Decision theory and artificial intelligence: 1. A semantics-based region analyzer," *Artificial Intelligence 5*, 4, 349-371, 1974.

Hanson, A.R. and E.M. Riseman, "Segmentation of natural scenes," in A.R. Hanson and E.M. Riseman (Eds). *Computer Vision Systems.* NY: Academic Press, 1978.

Haralick, R.M. and G.L. Elliott, "Increasing tree search efficiency for constraint satisfaction problems," *Proc.*, 6th IJCAI, Tokyo, 356-364, August 1979.

Hinton, G.E., unpublished report, U. California at San Diego, 1980.

Hinton, G.E., "Relaxation and its role in vision," Ph.D. dissertation, U. Edinburgh, 1979.

Horn, B.K.P and B.G. Schunck, "Determining optical flow," AI Memo 572, AI Lab, MIT, April 1980.

Horn, B.K.P. and R.W. Sjoberg, "Calculating the reflectance map," *Proc.*, DARPA IU Workshop, 115-126, Pittsburgh, PA, November 1978.

Hough, P.V.C., "Method and means for recognizing complex patterns," U.S. Patent 3,069,654, 1962.

Huffman, D.A., "Impossible objects as nonsense sentences," in B. Meltzer and D. Michie (Eds). *Machine Intelligence 6.* Edinburgh: Edinburgh U. Press, 1971.

Hummel, R. and S. Zucker, "On the foundations of relaxation labeling processes," TR, Dept. of Electrical Engineering, McGill U., 1980.

Hurvich, L.M. and D. Jameson, "An opponent-process theory of color vision," *Psych Review 64,* 384-390, 1957.

Ikeuchi, K., "Numerical shape from shading and occluding contours in a single view," AI Memo 566, AI Lab, MIT, revised February 1980.

Kanade, T., "Recovery of the three-dimensional shape of an object from a single view," CMU-CS-79-153, Computer Science Dept, Carnegie-Mellon U., October 1979.

Kanade, T., "A theory of Origami world," CMU-CS-78-144, Computer Science Dept, Carnegie-Mellon U., 1978.

Kender, J.R., "Shape from texture: A brief overview and a new aggregation transform," *Proc.*, DARPA IU Workshop, 79-84, Pittsburgh, PA, November 1978.

Kimme, C., D.H. Ballard, and J. Sklansky, "Finding circles by an array of accumulators," *CACM 18,* 1, 120-122, February 1975.

Marr, D., "Representing visual information," in A.R. Hanson and E.M. Riseman (Eds). *Computer Vision Systems.* NY: Academic Press, 1978.

Marr, D., "Representing and computing visual information," in P.H. Winston and R.H. Brown (Eds). *Artificial Intelligence: An MIT Perspective.* Cambridge, MA: The MIT Press, 1979.

Marr, D. and T. Poggio, "Cooperative computation of stereo disparity," *Science 194,* 283-287, 1976.

Nahin, P.J., "The theory and measurement of a silhouette descriptor for image preprocessing and recognition," *Pattern Recognition 6,* 2, October 1974.

Ohlander, R., K. Price, and D.R. Reddy, "Picture segmentation using a recursive region splitting method," *CGIP 8,* 3, December 1979.

Prager, J.M., "Extracting and labeling boundary segments in natural scenes," *IEEE Trans. PAMI 2,* 1, 16-27, January 1980.

Rosenfeld, A., R.A. Hummel, and S.W. Zucker, "Scene labelling by relaxation operations," *IEEE Trans. SMC 6,* 1976.

Sabbah, D., "Design of a highly parallel visual recognition system," Computer Science Dept, U. Rochester; submitted to 7th IJCAI, 1981.

Shapiro, S.D., "Generalization of the Hough transform for curve detection in noisy digital images," *Proc.*, 4th IJCPR, Kyoto, Japan, 710-714, November 1978.

Sloan, K.R., Jr. and D.H. Ballard, "Experience with the generalized Hough transform," *Proc.*, 5th Int'l Conf on Pattern Recognition, Miami Beach, FL, December 1980.

Tanimoto, S. and T. Pavlidis, "A hierarchical data structure for picture processing," *CGIP 4,* 2, 104-

119, June 1975.

Ullman, S., "Interpretation of visual motion," Ph.D. dissertation, MIT, 1977.

Ullman, S., "Relaxation and constrained optimization by local processes," *CGIP 10*, 115-125, 1979.

Waltz, D.I., "Generating semantic descriptions from drawings of scenes with shadows," Ph.D. dissertation, AI Lab, MIT, 1972; also in P.H. Winston (Ed). *The Psychology of Computer Vision*. New York: McGraw-Hill, 1975.

Zucker, S.W., "Labeling lines and links: An experiment in cooperative computation," TR 80-5, Computer Vision and Graphics Lab., McGill U., February 1980.

Zucker, S.W., "Relaxation labelling and the reduction of local ambiguities," TR 451, Computer Science Dept, U. Maryland, 1976.

Acknowledgements
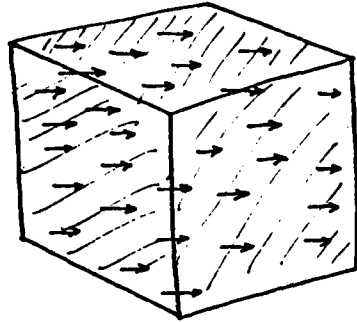
a.
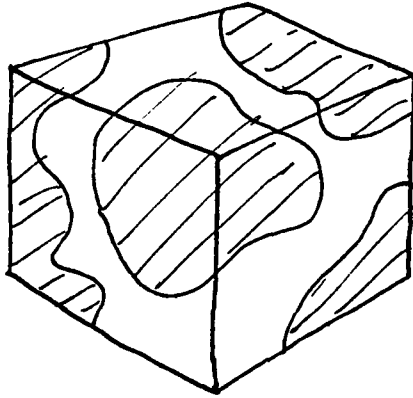
b.
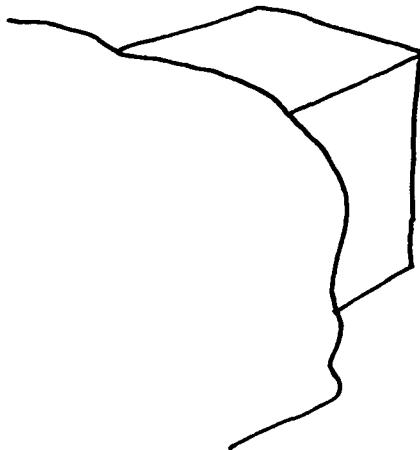
c.

Figure 1.

Isolated edge
⇒ low confidence

edge with aligned
neighbors ⇒ high confidence

Figure 2
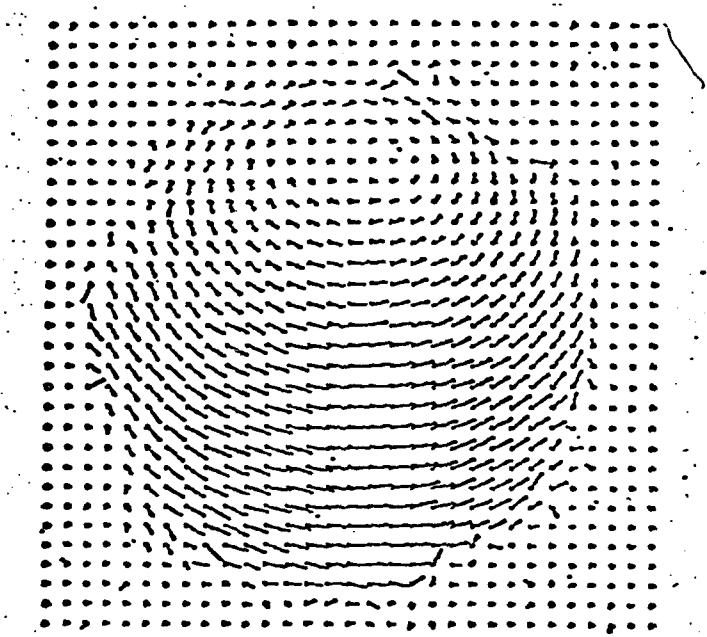
Figure 3 Horn's rotating sphere
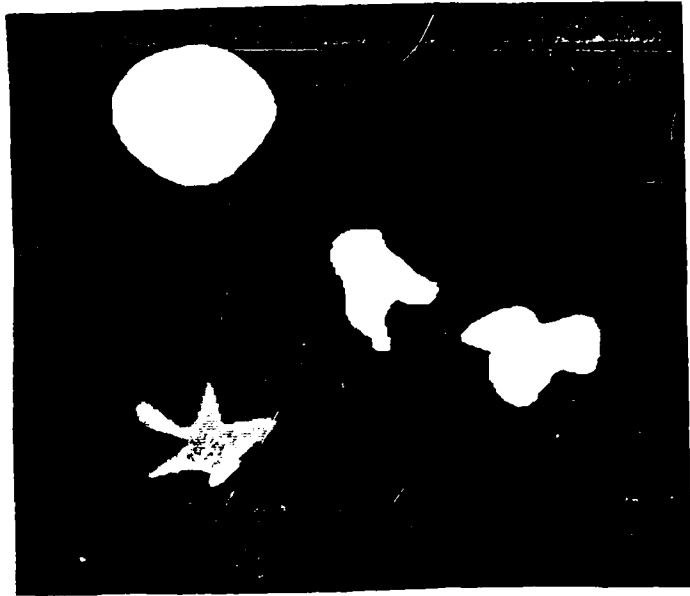
a.

b.
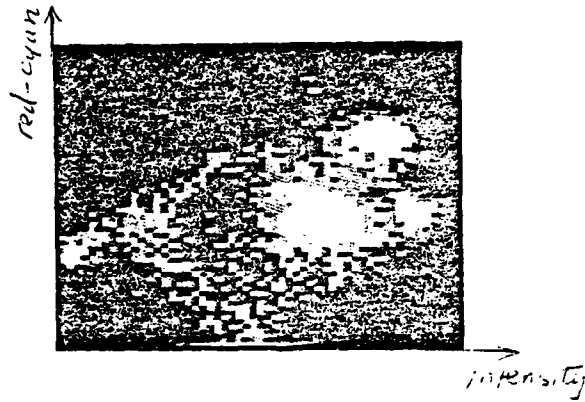
Figure 4. Hanson and Riseman
Color Segmentation
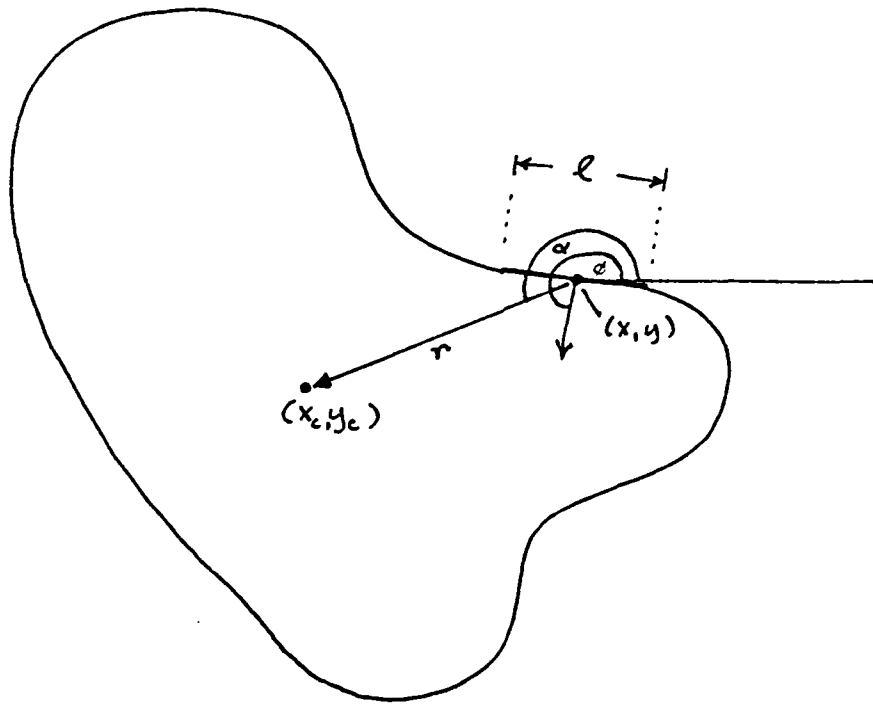
Figure 5   Geometry for the Shape
Hough Transform

a.

b.

Figure 6

```
┌─────────────────────────────────────────┐
│                                         │
│            O B J E C T S                │
│                                         │
├────────────────────┬────────────────────┤
│                    │                    │
│      COMPLEX       │    P L A N E S     │
│      JOINTS        │                    │
│                    │                    │
├────────────────────┴────────────────────┤
│                                         │
│            L J O I N T S                │
│                                         │
├────────────────────┬────────────────────┤
│                    │                    │
│     R A Y S        │    L I N E S       │
│                    │                    │
├────────────────────┴────────────────────┤
│                                         │
│            E D G E S                    │
│                                         │
└─────────────────────────────────────────┘
```
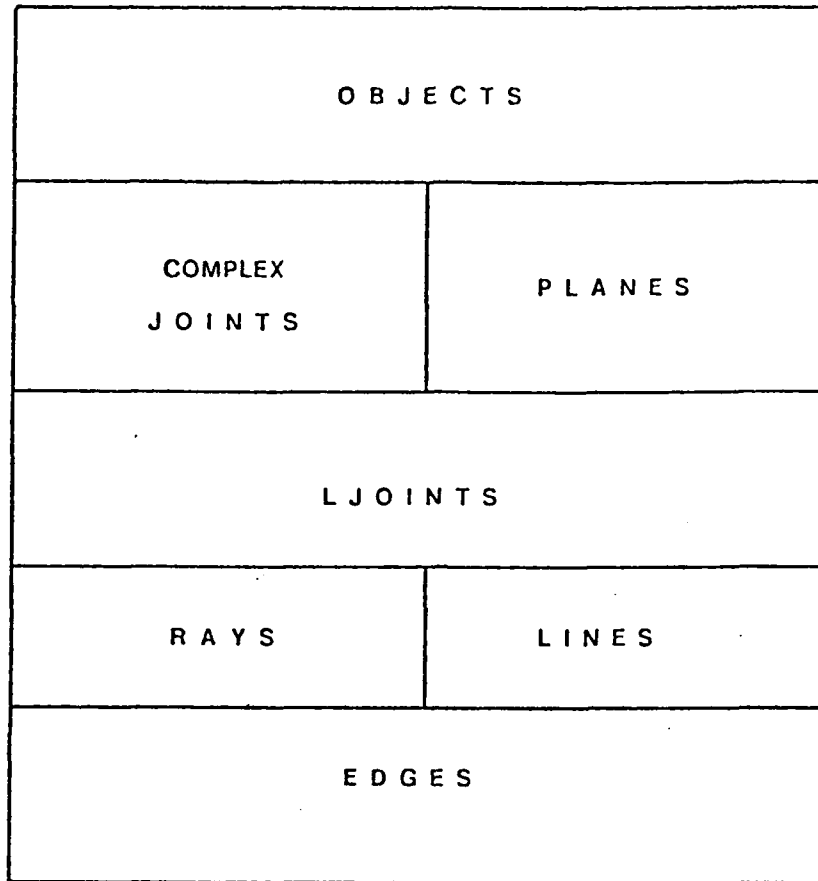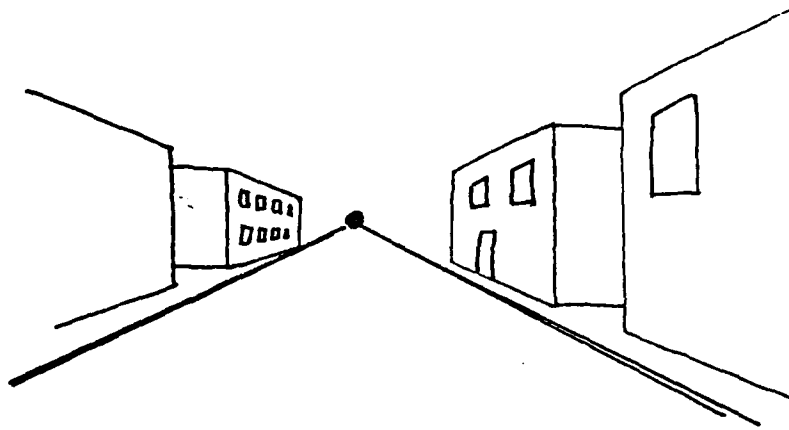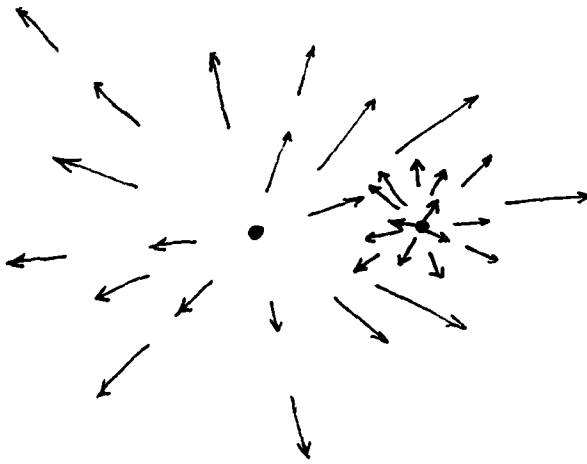
Figure 7. Sabbah's origami world hierarchy
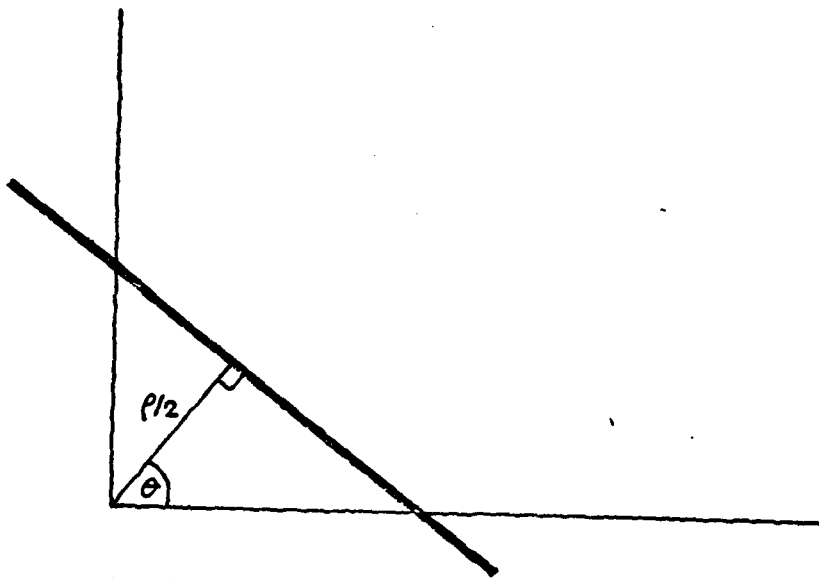
a. Vanishing Point

b. Radial Flow; Two FOEs

Figure 8
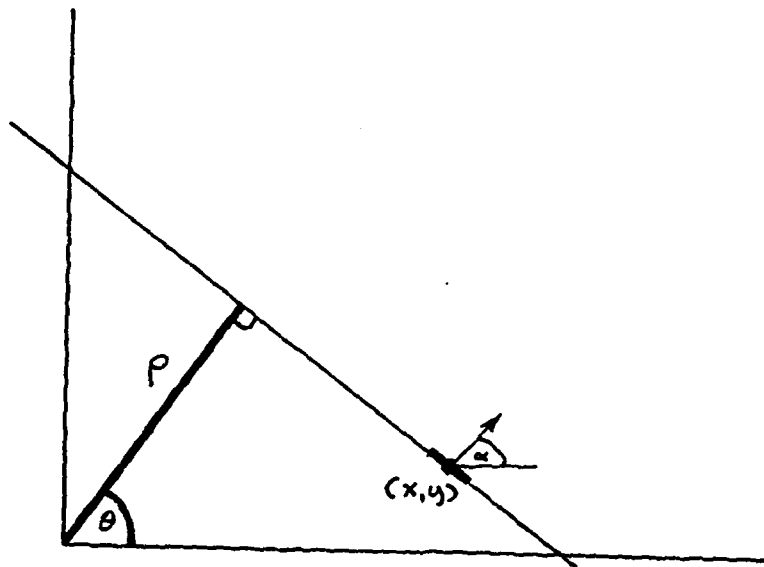
a. Lower Level : Colinear edge elements
→ lines

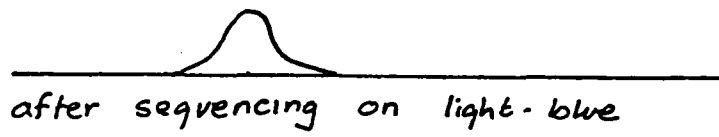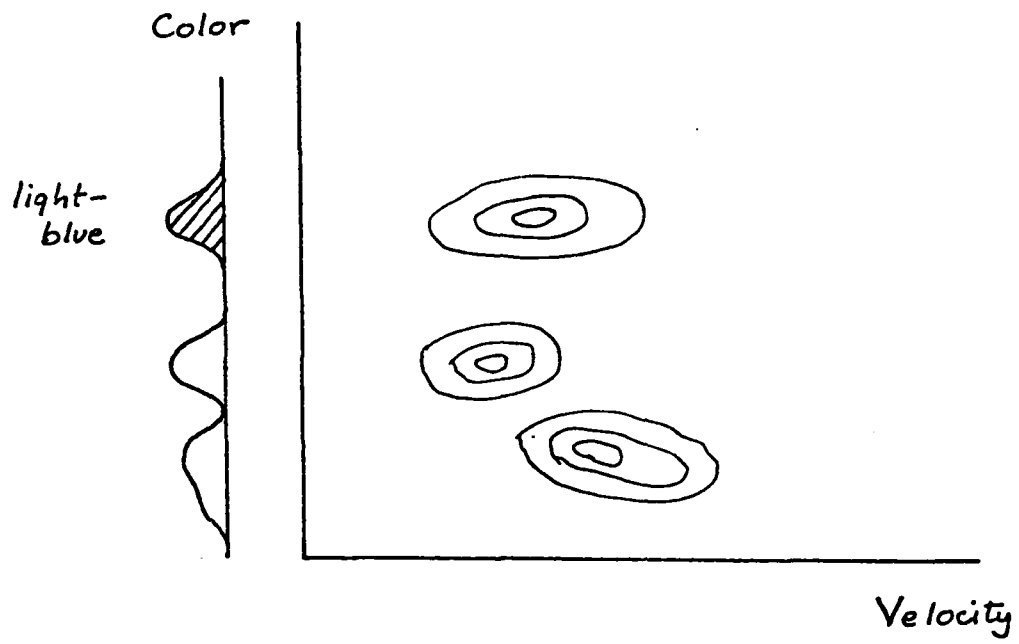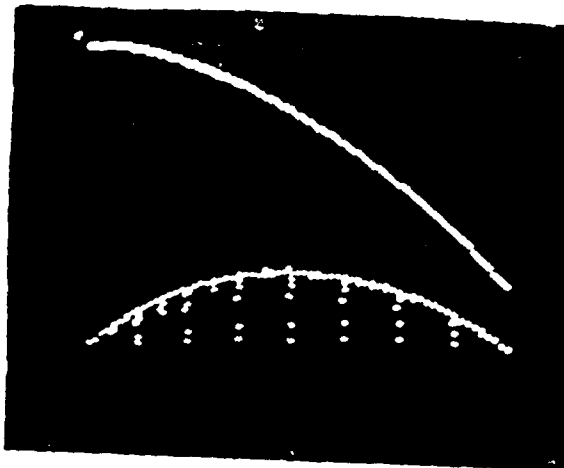b. Higher Level : Circles in line space
→ points

Figure 9

Color

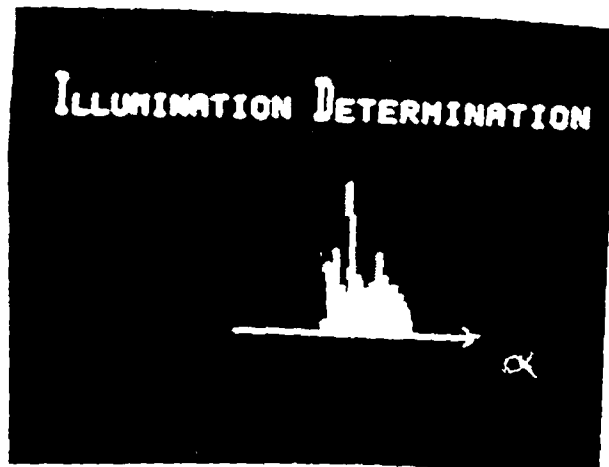light-
blue

Velocity

before sequencing on light-blue

after sequencing on light-blue

Figure 10

a.

b.

ILLUMINATION DETERMINATION

c.

α
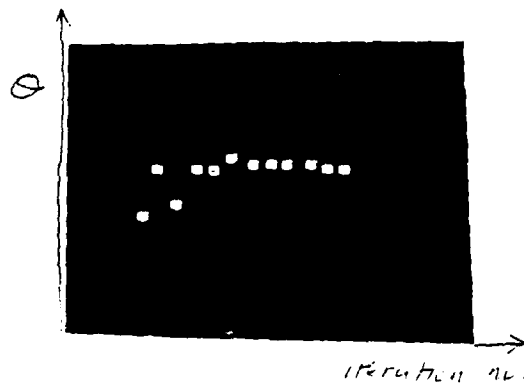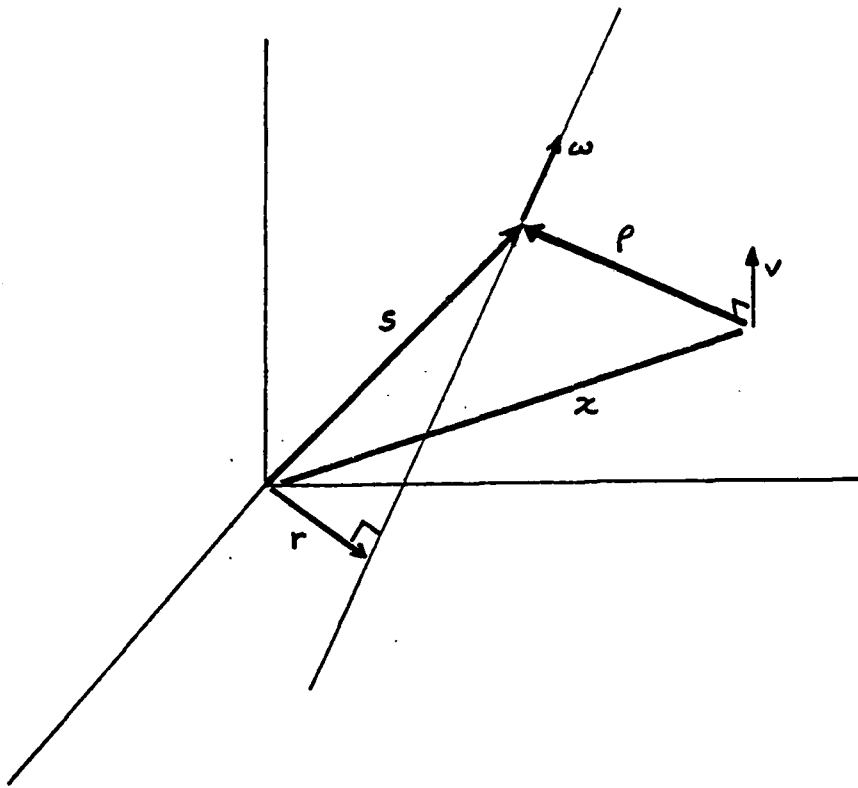
Θ

d.

Iteration n.

Figure 11

Figure 12 Geometry for Rigid-body Motion

END

DATE
FILMED

7-8|1

DTIC