

Representation and Control in Vision

Takeo Kanade

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pa. 15213

1. Introduction

One of the central issues in vision is how to represent and use knowledge relevant to understanding the image. Partly because vision is so difficult, and partly because even the cheapest solutions can still be so useful, approaches to vision problems have had a tendency to be *ad hoc* and heuristic. Recently, however, new thrusts in computer vision are emerging, most notably in the Image Understanding community, that try to pursue more systematic and computational approaches. [1]

This article attempts to introduce such new Image Understanding approaches to vision. It first presents the author's view of structure of vision: what types of information must be dealt with and what levels of knowledge are involved in transforming one type of information into another. Then, representative research progress in Image Understanding is reviewed which addresses use and representation of knowledge in vision. Specifically, we will discuss:

- e Formalization of physical knowledge into computational forms
- e Use of 3D models
- e Construction of scene descriptions from images
- e Organization and control of vision systems

2. Structure of Vision

Let us consider what vision really is. Vision involves visual sensing and interpretation. Visual sensing is a *projection* of a physical environment into a form of representation called *images*. Projection can vary from the most ordinary picture taken by cameras to active sensing such as by laser range finders. Images can also range from a single-point light flux measurement to 3-dimensional range data. Then vision is defined as a process of understanding the environment through the projected images, or in short *inverse projection*.

The vision process must involve various types of information. Figure 1 is a schematic diagram showing one possible dichotomy of such types and their primary relationships. [14] The general idea is as follows: given an image, cues (*picture domain cues* or *scene domain cues*) are extracted; these are then used to access the *model* of the task world and to instantiate it; the *instantiated model* is verified by projecting it back to the picture level (*picture interpretation*) and by matching it with the input image. This positive feedback loop is the basic drive in the analysis process.

Several points should be noted about this model. First of all, there is an important distinction between the picture domain and the scene domain. In short, a *picture* is a projection of a *scene*. Thus the picture domain cues are the features observed in the picture, such as line segments, homogeneous regions, intensity gradients, etc. The scene domain cues are the features which are the cause of the picture domain cues, such as edge configurations, surface orientations, reflectances, volumes etc. This distinction prevents one from confusing features in the picture domain with those in the scene domain. For example, the "above" or "adjacent" relationship between regions in the picture does not necessarily correspond to the "on" or "next-to" relation between objects. Though the most basic and important scene domain cues are spatial three-dimensional (3-D) configurations, there are other important ones, such as reflectances and lighting conditions. The model contains the generic descriptions of the task world. It can be thought of as containing information about the generic shapes of objects and relations among them. Note that generic

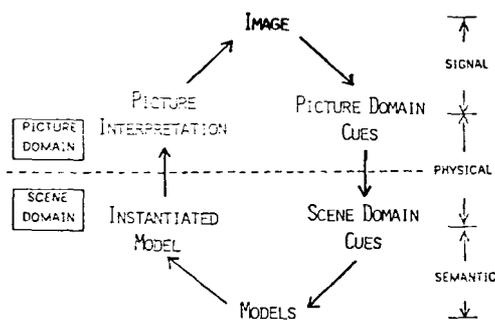


Figure 1: A model of vision process. Roughly speaking, the right half of the cycle corresponds to the data-driven, bottom-up, hypothesis-generation phase, and the left half to the model-driven, top-down, verification phase of vision.

models are to be represented in terms of scene domain cues. The instantiated model is the specific situation being depicted in the picture: the specific objects, their sizes, their placement, etc. The picture interpretation is a projection (an anticipated picture) of the instantiated model under the specific conditions (view direction, lighting conditions, etc.). It gives an explanation of the image by the descriptions such as what lines and regions of what properties *should* appear in the image, what set of lines or regions constitute the objects, and so on. An example of this type of information is an image segmented into regions with semantic labels attached to each. It is different from an *image* which is only a collection of image features, such as color and intensity.

The model of Figure 1 involves three levels of knowledge -- signal, physical, and semantic. Roughly speaking, the *signal level* involves manipulating image features as 2-D patterns, the *physical level* involves bridging the picture and scene domains so that image features can be interpreted as scene features, and the *semantic level* involves working in the scene domain to exploit the task world constraints.

The difference among these levels can be illustrated by the following example. Suppose that a histogram of the gray levels of an image shows a clear bimodality. The value at the valley is often selected as a good threshold. This is based on the *signal level* knowledge that in such a case often two sources of monomodal distribution are present. Therefore, a threshold value at that valley will give a good splitting of the image. We are also assuming that a salt-and-pepper image (which can produce the same histogram) is rare. This signal-level histogramming technique is actually based on *physical level* knowledge. We know that two surfaces, whose reflectivity characteristics are different from each other but constant within each, or which have different orientations, can produce two regions in the image whose gray levels are different but relatively constant within each. Why is it significant to extract those surfaces for understanding the image? Because we know that surfaces with different reflectance or orientations are the *semantically* meaningful units which constitute the objects in our world.

As another example, the extraction of straight lines by detecting sudden changes of intensity on the signal level is justified by the physical-level observation that such intensity changes are caused by sudden change of orientations (convexity and concavity), distance (occlusion), illumination (shadow), and reflectance (material change), all of which are usually important clues in the semantic analysis of the scene content.

It is *not* meant that the analysis should always follow the routes shown in Figure 1. In fact, one can *combine* several steps into one, or precompile necessary knowledge about some entities into a processing procedure, so that those entities are not explicitly treated. However, the meaning and limitations of many methods can be illustrated by considering what types of knowledge are *incorporated* into them, according to this paradigm.

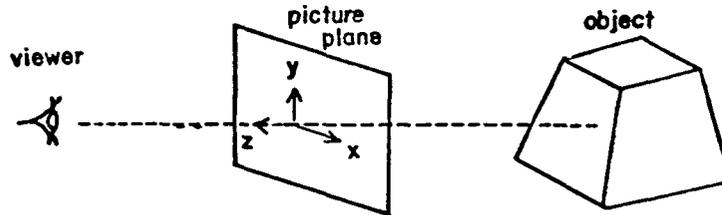
Notice that essential difficulties in vision lie in crossing the boundary between the scene and picture domains from picture to scene: it is *ambiguous*, thus additional constraints are to be used. Generality of vision depends on how to derive and use necessary constraints.

3. Converting Physical Level of Knowledge into Computational Constraints

Historically, there are several annoying image phenomena which often cause vision programs to fail in image analysis: they include deformed shape due to slanted views, shadings due to curved surfaces and lighting, and textural patterns, and shadow. That may be still true in most applications, but it has begun to be understood that these are rich sources of information about object shape. An interesting class of theories have been developed to handle these sources. They are all for relating picture domain cues with scene domain cues so that they can be used in recovering 3-dimensional scene features from images. This section will briefly introduce such theories which allow computational usage of constraints from image intensity and geometry in order to exploit 2D-shape, shading, texture, and shadow in the image.

Since we will use *gradient space* [32] [20] throughout the section as a convenient tool to represent surface orientations, let us first define it. Our coordinate system x - y - z is placed so that the optical axis is the z axis and the image plane is the x - y plane. (See Figure 2.) Consider a surface,

$$-z = f(x,y) \quad (1)$$



$$\text{planes: } -z = px + qy + c \longrightarrow \text{point: } (p, q)$$

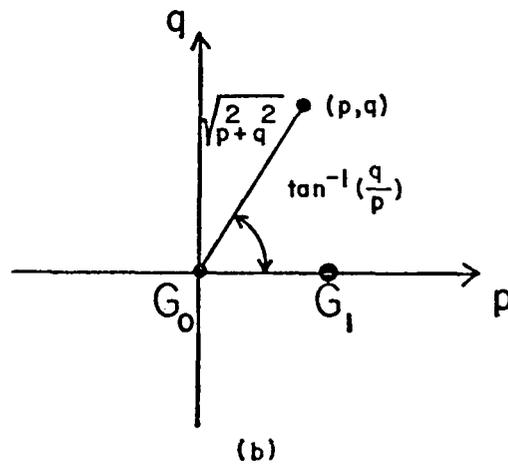


Figure 2: (a) Imaging geometry including the object, the picture, and the viewer. (b) mapping of planes to a gradient.

The gradient space is defined by (p,q) where

$$p = \frac{\partial f}{\partial x}, \quad q = \frac{\partial f}{\partial y} \tag{2}$$

That is, p and q are the rate of change in depth on the surface along the x and y direction. We can easily see that $(p,q,1)$ has the direction of the surface normal.

Gradients are constant over a planar surface, and the gradient (p,q) corresponds to a set of parallel planes:

$$-z = px + qy + c \tag{3}$$

where c is arbitrary. Throughout this section we will assume orthographic projection for simplicity rather than perspective projection.

Under orthographic projection, the gradients of planes and the image line on the image have an interesting relationship. Referring to Figure 3, let two planes P_1 and P_2 intersect in the space and let the intersection edge be depicted as a line l in the image. Then the line in the gradient space connecting the corresponding gradients $G_1 = (p_1, q_1)$ is perpendicular to the line l . This can be seen in the following way. The normals of the two surfaces have directions $n_1 = (p_1, q_1, 1)$ and $n_2 = (p_2, q_2, 1)$. Their cross product $n_1 \times n_2$ represents the direction of the intersection edge in the space. Since we assume orthographic projection the direction of the line l in the image is given by the x and y components of this cross product: i.e., $(q_1 - q_2, p_2 - p_1)$. This vector is obviously perpendicular to the vector $G_1 - G_2 = (p_1 - p_2, q_1 - q_2)$ which connects G_1 and G_2 . Moreover if the intersection edge is convex viewed from the viewer, then the positional order of G_1 and G_2 is the same as the regions in the picture corresponding to P_1 and P_2 ; if concave, the order is reversed.

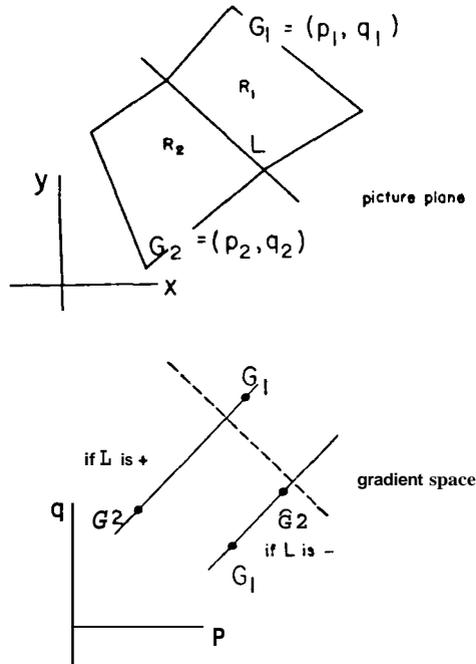


Figure 3: Properties of dual lines. If two planes meet and the intersection line is projected as a picture line L , then the gradients of the two planes are on a gradient-space line which is perpendicular to L .

3.1. Image Intensity

3.1 - Shading

Grey-level shading has been neglected or even thrown out in most robotics vision, though it is well known that it gives information on surface curvature. This is partly due to the dominance of binary image processing for fast processing and partly due to lack of adequate theory and techniques for exploiting shading information. Horn [9] did a pioneering work in shape-from-shading theory. Figure 4 shows a simple model of imaging: it consists of a point light source, a surface patch, and a viewer. In general the intensity in the image corresponding to the surface patch is a complex function of illumination position, surface material, surface position, surface orientation, viewer position, etc. Let us assume, for simplicity, a Lambertian surface (ie, perfectly uniform diffuse reflection) and a parallel incident light (ie, a distance point source). The observed intensity I which corresponds to the surface patch is given by,

$$I = I_0 \rho \cos(i) \quad (4)$$

where I_0 is the intensity of the incident light, ρ the reflectance of the surface, and i the incident angle which is made by the incident light and the surface normal.

In order for this equation to be usable in vision, we need to convert it into the imaging coordinate frame. This can be done by using the gradient space. The surface normal of the patch is $\mathbf{n} = (p, q, 1)$ and the direction vector of the illumination is expressed as $\mathbf{s} = (p_s, q_s, 1)$. Since $\cos(i) = \mathbf{n} \cdot \mathbf{s} / \|\mathbf{n}\| \|\mathbf{s}\|$,

$$I(x, y) = I_0 \rho (p p_s + q q_s + 1) / \sqrt{p^2 + q^2 + 1} \sqrt{p_s^2 + q_s^2 + 1} = R(p, q) \quad (5)$$

In this way, the observed intensity $I(x, y)$ at an image point (x, y) is expressed as a function $R(p, q)$ of the surface orientation (p, q) of the corresponding surface point. Figure 5 shows the loci of equal-intensity gradients $R(p, q) = I$, for the case $(p_s, q_s, 1) = (0.7, 0.3, 1)$; i.e., the light comes from slightly above and right of the viewing angle. Figure 5 is called a reflectance map.

It is worthwhile to note the significance of the representation like the reflectance map (or equation (5)) as compared with equation (4). Equation (4) represents the physical rule in imaging as viewed from a third person who observes the imaging process. In contrast, equation (5), though representing the same rule, is in the viewer's coordinate frame. This is essential for using the rule to interpret the images because the images are recorded in that coordinate frame.

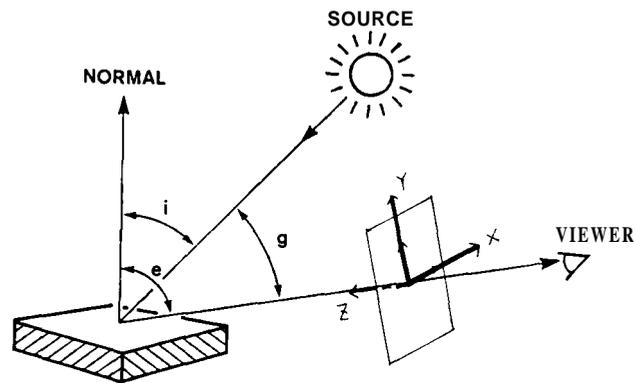


Figure 4: A simple reflection and imaging model

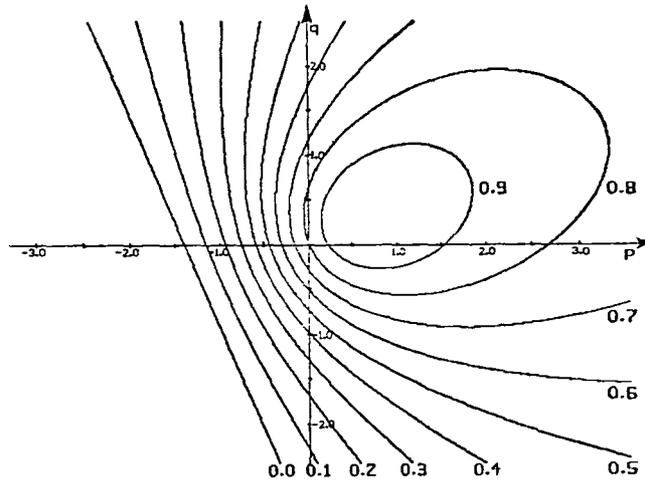


Figure 5: A reflectance map for a Lambertian surface illuminated by parallel flux of light from the direction $(0.7, 0.3, 1)$.

The reflectance maps can be used in a few ways. One is for shape-from-shading: given an image of an object, compute the shape of the object using the shading information. This can be accomplished by assigning to each pixel (i, j) in the image a gradient (p_{ij}, q_{ij}) . The reflectance map alone does not give enough constraints to uniquely determine the gradient at each point: i.e., $R^{-1}: I \rightarrow (p, q)$ is not a unique function. Here we need additional constraints. Ikcuchi and Horn [12] demonstrated a numerical shape from shading by assuming surface smoothness. They obtained the solution by minimizing the following.

$$E = \sum_i \sum_j F_{ij} = \sum_i \sum_j \{ \{ I(i, j) - R(p_{ij}, q_{ij}) \}^2 + \lambda^2 \{ (p_{ij} - \bar{p}_{ij})^2 + (q_{ij} - \bar{q}_{ij})^2 \} \} \quad (6)$$

The first term in E , corresponds to the difference between the observed and expected intensities. The second term approximates the Laplacian of the surface orientation and thus provides index of non-smoothness. In order to solve this minimization we further need boundary conditions: certain places in the image where the surface orientations are known. Occluding contours provide such boundary conditions. The surface orientations of those pixels at the occluding contours are known because the surface normal there is perpendicular to the line of sight (z axis) and the occluding contour projected onto the image. Using all these constraints 3D shape recovery from a grey-level image was demonstrated.

Another use of the reflectance map is photometric stereo [35]. Multiple images of an object are taken by the camera at one position, but with different lighting directions. At each pixel position, each image provides constraints on surface orientation according to the reflectance map, which corresponds to that lighting condition. Three or more images can determine the surface orientations uniquely. This method eliminates the so called correspondence problem in stereo, because the images are taken from a single position and thus are registered. It suggests use of dynamic clever control in lighting to inspect surface texture and anomalies in such applications as soldering and casting inspection.

The reflection properties of materials are complicated, and rather than analytical forms, they are often measured in forms such as the bidirectional reflectance distribution function (BRDF) or goniophotometric measurement. There have been developed computational procedures to convert such data into reflectance maps [10][30]. These techniques make it possible to apply the reflectance map method to robotics tasks, because the environments are usually constrained and surface properties are known or measurable.

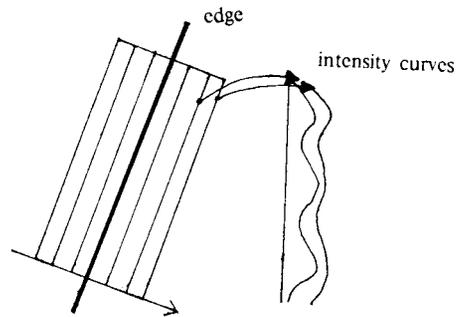


Figure 6: A stripe region around an edge, in which columns of intensity curves are taken for correlation.

3.1.2. Classification of Edge Types

We have noted that so called image edges (discontinuity of image intensity) can correspond to many types of scene edges. The most representative types are:

Type	Discontinuous Scene Feature
Convex/Concave	Surface orientation (Continuous distance)
Occluding	Distance (Different surfaces)
Cast Shadow	Illumination (Same surface)
Self Shadow	Illumination derivative
Pattern	Reflectance

If we can classify edges in the image into these types, it will make it far easier to analyze the 3-dimensional shapes of objects from the image. For example, the object shape can be found from occluding boundaries without being confused by non-shape related edges; and stereo matching would be much easier because the edge types tell what kind of different appearance should be expected on either side of an edge in making correspondence.

Initially, use of edge profiles [9] or intensity derivatives near the edge [3] was suggested for edge classification based on consideration of micro structures near the edge. Recently Fischler and Witkin [7] studied an interesting method to detect occluding boundaries and cast shadow edges in the image. Consider strip regions along an edge as shown in Figure 6 and take columns of intensity curves taken parallel with the edge along its length. A sequence of linear regressions is performed between each pair of consecutive intensity curves. As a result a normalized correlation coefficient is obtained, together with additive and multiplicative regression terms, each as a function of the location across the edge. An occluding boundary is indicated by a sharp notch in an otherwise high correlation at the nominal edge location. A cast shadow boundary is indicated by high correlation maintained across the edge with a sharp spike or notch in the additive and multiplicative regression term due to transition between shadow and nonshadow sides.

Justification of this method is given by the following simplified arguments. At the occluding boundary we are looking at different surfaces on each side. Thus the image properties across it should not be correlated. At the shadow boundary, illumination changes but the surface is the same. Thus image intensities across it should be correlated with change only in either magnitude or scale. This classification method rests on the principle that the coherence in the image intensity reflects the real coherence in the scene.

3.2. Geometry

Geometry is another important physical level of knowledge which governs the vision process and from which constraints can be extracted.

3.2.1. 2D Shape

Two dimensional shapes of regions (i.e., projection of surfaces) in the image convey information about the three-dimensional shape of an object in the scene. This is illustrated by such simple drawings in Figure 7. Figures 7(a) and (b) are topologically the same and the slight difference in the 2D shape of the lower two regions results in different perceptions "cube" and "trapezoidal block". In fact, even such simple figures as Figures 7 (c) and (d) already invoke the perception of surface orientations. Certain geometrical properties should be the source of that perception. These phenomena may have been often studied in their psychological aspects, but until recently very little has been formulated in a manner usable for machine vision.

Kanade [15] demonstrated a systematic method to recover 3-dimensional shapes from a single view by mapping image properties into shape constraints. The 2D shape property of Figure 7 (c) and (d) is one of the properties he studied. It was named *skewed symmetry*, because they are interpreted as symmetric figures viewed obliquely. In other words, whereas in the usual symmetry reflective correspondences are found along the direction perpendicular to the symmetry axis, in the skewed symmetry it is found along the direction not necessarily perpendicular to the axis, but at a fixed angle to it. Formally, such shapes can be defined as 2-D Affine transforms of real symmetries.

There are a good body of psychological experiments which suggests that human observers can perceive surface orientations from figures with this property. This is probably because such qualitative symmetry in the image is often due to real symmetry in the scene.

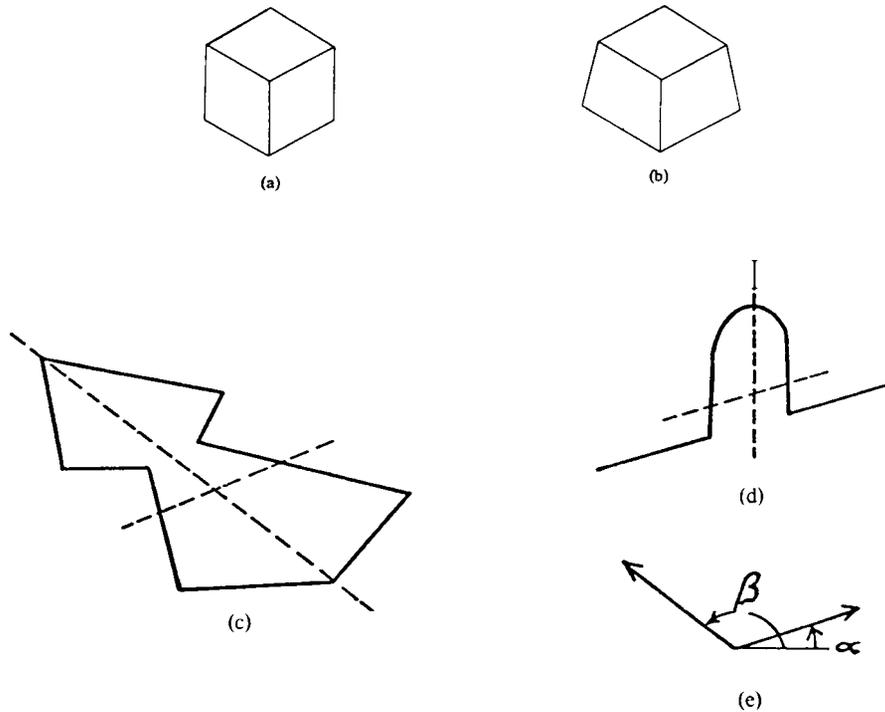


Figure 7: (a)(b) Simple line drawings: (a) "cube"; (b) "trapezoidal block"; (c)(d) skewed symmetry; (e) axes of skewed symmetry of (c)

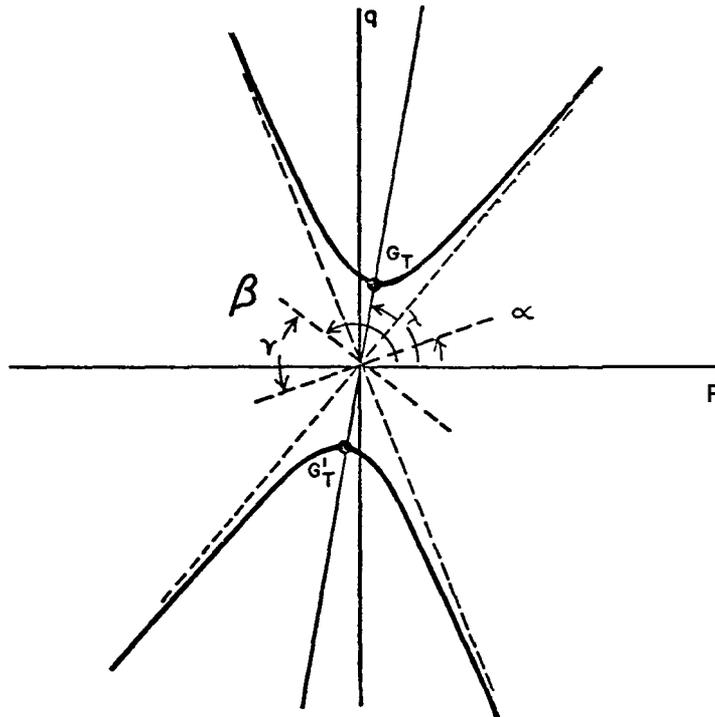


Figure 8: The hyperbola of eq. (8) corresponding to Figure 7(c). The axis of the hyperbola is the bisector of the obtuse angle made by α and β . The asymptotes make the same angle as the acute angle made by α and β . The tips or vertices G_T and G'_T of the hyperbola represent special orientations with interesting properties. Especially, since they are closest to the origin of the gradient space, and since in general the distance from the origin to a gradient (p,q) represents the magnitude of the surface slant G , and G_T and G'_T correspond to the least slanted orientations that can produce the skewed symmetry in the picture from a real symmetry in the scene.

Now let us associate the following assumption with this image property: *"A skewed symmetry depicts a real symmetry viewed from some unknown view angle."* Note that the converse of this assumption is always true in orthographic projection. As shown in Figure 7(c), a skewed symmetry defines two directions: let us call them the skewed-symmetry axis and the skewed-transverse axis, and denote their directional angles in the picture by α and β , respectively. Let $G = (p,q)$ be the gradient of the plane which includes the skewed symmetry. In general the 2-D unit vector e in the direction γ is $e = (\cos\gamma, \sin\gamma)$. From (3), the 3-D vector corresponding to e on the plane in the space is given as:

$$u_\gamma = (\cos\gamma, \sin\gamma, -(p\cos\gamma + q\sin\gamma)). \quad (7)$$

The assumption about the skewed symmetry demands that the two 3D vectors u_α and u_β be perpendicular in the 3D space: i.e., their inner product vanishes, $u_\alpha \cdot u_\beta = 0$, or:

$$\cos(\alpha - \beta) + (p\cos\alpha + q\sin\alpha)(p\cos\beta + q\sin\beta) = 0. \quad (8)$$

By rotating the p - q coordinates by the amount $\lambda = (\alpha + \beta)/2$ into the p' - q' coordinates so that the new p' - q' axes

are the bisects of *the* angle made by the skewed-symmetry and skewed- the skewed-symmetry and skewed-transverse axes, it is easy to show that (8) represents a hyperbola in the gradient space shown in Figure 8. That is, the skewed symmetry defined by \mathbf{a} and \mathbf{b} in the picture can be a projection of a real symmetry *if and only if* its surface gradient (p,q) is on this hyperbola. The skewed symmetry thus imposes a one-dimensional family of constraints on the underlying surface orientation. Figure 9 illustrates how this skewed symmetry constraint can be used to recover the shape of "cube" from the image.

The same approach was extended to other properties: parallel lines, affine-transformable patterns, and textures [17]. We can summarize the assumptions used in these cases: *regular properties observable in the picture are not by accident, but are projections of some preferred corresponding 3-D properties*. Figure 10 lists instances of this principle of *non-accidental image regularity*. Note also that the principle used in classifying the edge types in the previous section is also the same effect: the coherence in the image reflects the real coherence. Since the mapping from the picture domain to the Scene domain is one-to-many (ambiguous), we need to rely on this type of general assumptions or task-specific constraints to resolve the ambiguity.

3.2.2 Shadow

Shadow gives good clues on spatial relationship between objects and surfaces. Aerial photo interpreters make much use of it in figuring out, for example, the height of the objects. Lowe and Binford [19] demonstrated the reconstructing of the shape of an airplane by using shadow information. They first paired shadow-making edges and casted-shadow edges on the ground by assuming the ground plane and the sun angle. Then the height of the airplane along the contour made of shadow-making edges was estimated, which gave the shape of the airplane.

Shafer and Kanade [31] investigated a general and compact gradient-space representation of geometrical constraints given by shadow. Figure 11 shows a basic shadow problem. It consists of the parallel-light illuminator I , the shadow making (occluding) surface S_o and the shadow surface S_s . The problem includes six parameters to be computed: the gradient $G_o = (p_o, q_o)$ of S_o , the gradient of $G_s = (p_s, q_s)$ of S_s and the direction of illumination (p_l, q_l) .

This problem can be studied by considering two other surfaces S_{11} and S_{12} (and their gradients G_{11} and G_{12}), each of which includes a pair of shadow making and casted shadow edges: for example, F_{o1} and F_{s1} define S_{11} . Note that S_{11} and S_s make a concave edge along F_{s1} , and so do S_{12} and S_s along F_{s2} . Now three constraints are provided from the basic shadow problem geometry: 1) The angle $G_o - G_{11} - G_s$, which comes from the angle $F_{o1} - F_{s1}$; 2) The angle

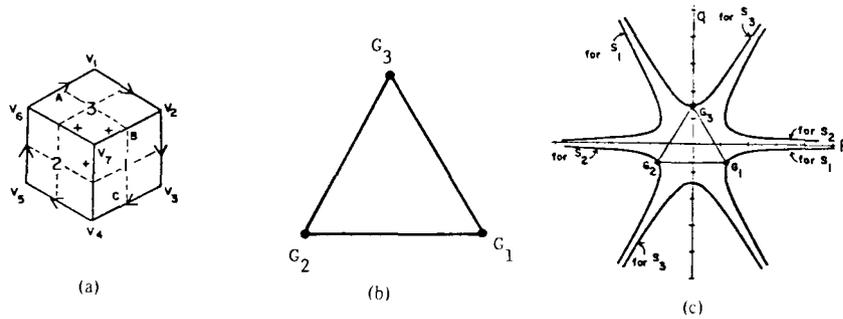


Figure 9: (a) A labeled line drawing: the dotted lines show the axes of skewed symmetry. (b) The constraints on the gradients of the three surfaces due to their interconnection: the gradients form a triangle in the gradient space whose shape and orientation should as shown, but the location and the size are arbitrary. (c) The hyperbolas shown correspond to the skewed symmetries of the three regions. The problem is thus how to place the triangle of (b), by translation and scale change, so that so that each vertex is on the corresponding hyperbola. The locations shown is proven to be the only position, and the resultant shape is a cube.

$G_0-G_{12}-G_s$, which comes from the angle between F_{02} and F_{s2} ; 3) The direction of the line l_{illum} (containing G_{11} and G_{12}), which comes from the direction of F_{11} (line containing the two vertices V_{012} and V_{s12}). We would therefore expect that three parameters must be given in advance, and the other three can be computed from the geometry. Figure 12 shows a construction for the case that the direction of illumination (actually the relative depth component of illumination vector - one parameter - is given) and the orientation of shadow casted surface (G_s) are known.

The basic shadow geometry provides three constraints, and thus three parameters have to be given by other means to solve the problem. It is interesting to compare this situation with the situation without shadows: an image which only depicts S_0 and S_s intersecting along F_{0s} . Here, there are four parameters (G_0 and G_s) to compute, and one constraint from the image (E), so three pieces of information are still needed in advance. With shadows, the same number of a priori parameters are needed, but one of them can be a description of the light source position instead of a description of a surface orientation. The geometrical significance of shadows is that they allow information about the light source to be used to solve the problem as a substitute for information about the surface orientations themselves.

3.3. Texture

Perception of depth and surface from texture gradient has been studied by psychologists. Recently, Kender [18] developed a very powerful computational paradigm for shape from texture. His central idea is a normalized texture property map (NTPM), which is again for mapping image features (texture in this case) into scene properties.

Let us show an example of NTPM for length. Suppose we have a texture pattern shown in Figure 13, which is made of line segments with two orientations: the horizontal ones with length L_2 and the diagonal ones (45°) with length L_1 . Consider in general a line segment in the image whose direction is of angle γ with the x axis and whose length is l . If that line segment is on a plane whose gradient is (p, q), what is the real length of the 3-dimensional line in the scene? From (7), $l \cdot u$ is the corresponding 3D vector, and its 3D length is

$$L(\gamma, l) = l|u| = l\sqrt{1 + (p\cos\gamma + q\sin\gamma)^2} \tag{9}$$

$L(\gamma, l)$ is the NTPM for a line segment with slope γ and length l . As with the reflectance map, we can represent (9) as a set of contours in the gradient space, $L(\gamma, l) = L$, each corresponding to such (p, q)'s that the 3D lines which are on the surface and which will be projected onto the image as a line of length l and angle γ are of length L . Figure 14 shows the NTPM for $\gamma = 45^\circ$. In general, the NTPM is a function of surface orientations, and its value represents a scene property (e.g., 3D line length) that the scene constituent in that surface orientation should take if the observed image property (e.g., 2D line length) comes from it. In other words, it represents *deprojected* scene properties from image.

<u>Picture</u>	<u>Scene</u>
Parallel in 2D	Parallel in 3D
Skewed symmetry	Symmetry
Similar color edge profile	Similar physical edge property
Matched T	Interception
Affine-transformable patterns	Similar or congruent patterns
Gradient in	Regularity in
--Spacing	Spacing--
--Length	Length--
--Density	Density--

Figure 10: Instances of the principle of non-accidental image regularity. Notice that the rules are mostly true when going from the scene to the picture, but the other direction is heuristic.

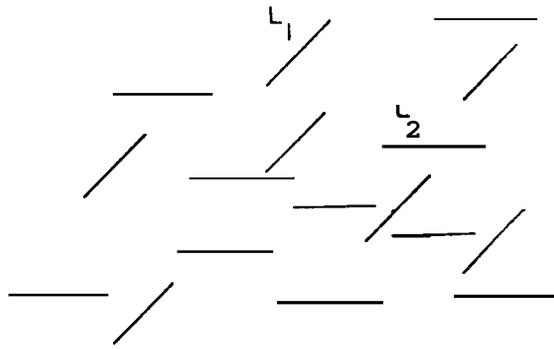


Figure 13: A texture image of line segments. It is assumed that all line segments are on a plane surface and their real length are the same. What orientations are possible?

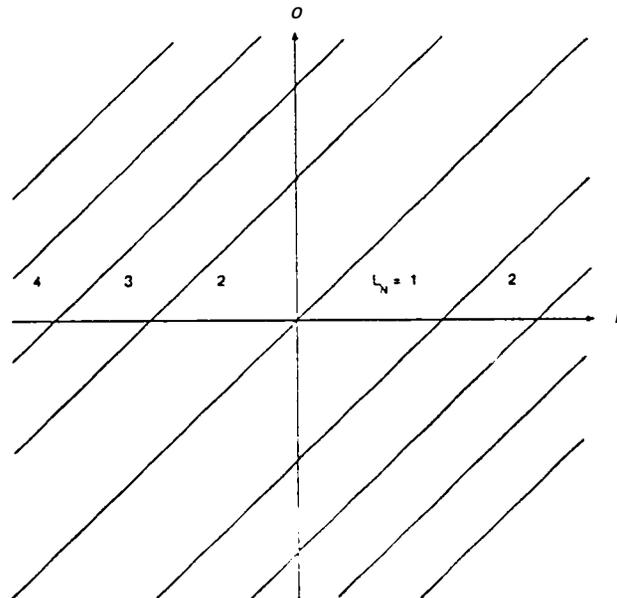


Figure 14: A normalized texture property map for length. $L(45^\circ, 1)$.

In Figure 13, if we assume that all the line segments are on the same plane and arc of the same length in the space, the possible orientations of the surface can be obtained by

$$L(0, L_2) = L(45^\circ, L_1) \quad (10)$$

or by graphically intersecting the NTPM surfaces. This is hyperbola in the gradient space. Though this is a very simple example, the same technique can be used to estimate the surface orientations of walls of buildings in outdoor scenes or of mechanical parts with textures.

We can observe a striking similarity between the case of texture and shading. We can make the correspondence between

<u>Shading</u>	<u>Texture</u>
pixel	texel
imaging set-up	scene constituent
reflectance map	NTPM

In fact, as in shape from shading, shape recovery of a curved surface which has a textured pattern on it can be performed by assuming surface smoothness and regularity in the texture. [17][11]

3.4. Remarks

The techniques and approaches in obtaining and representing constraints from the physical level of knowledge are suggestive to other areas of robotics which deal with sensing and interpreting natural environments. The key idea is to model the projection process and represent it in a form that can be used for inverse projection in conjunction with other constraints involved in interpretation. It is noteworthy that the forward projection rules used in Section 2 (reflection or projective geometry) are simple and fairly well understood. But they are often so local or microscopic that their direct application may result in gigantic unmanipulative equations. Appropriate representational spaces, such as gradient space, enable them to be applied in a macroscopic manner.

4. 3D Model-Based Vision

The theories presented in the previous section extract natural constraints under reasonable physical, mostly task independent assumptions. In this sense, those methods are model weak. Model-based vision attempts to use task-dependent semantic constraints. This idea is not new. In fact, ad hoc methods can be regarded as model based. Also two dimensional relationship among parts of objects, represented as either graphs or procedures, have been used to do direct two-dimensional pattern analysis such as face recognition, chest X-ray analysis and region segmentation of aerial photos. But what we aim at is 3D model-based vision which can cope with difficulties due to variations of object shapes and view angles. Although such perfect systems are probably still far away, we can see important progress.

An early attempt in model-based vision [6] used a predetermined set of models with fixed shapes to validate the hypothesized recognition result. Given an imperfect line drawing, the system extracted features and decided what objects appeared in what angles in the scene, then generated a line drawing anticipated if the recognition result was correct. It was compared with the input, and if they were close the recognition result was correct, otherwise other possibilities were pursued.

3-dimensional shape models can be also used to predict 2-dimensional appearances of objects beforehand under various lighting conditions. Then interpretation consists of mostly verification (*verification vision*). An example shown in Figure 15 is especially interesting because the appearances of stable postures of a part are computed from its 3-dimensional geometric model. [34] First, the convex hull of the object is obtained, and each of its surfaces is tested whether it provides a stable support of the object.

A general model-based vision system should have variable or generic models of objects and interpret an input image as instances of the generic model (recall Figure 1) to recognize the specific objects (e.g. size, shape) which appear

in a specific manner (e.g. orientation, distance). The basic difficulty here is the difference in the coordinate frames used in the generic models and in the representation obtained from images. The generic models are (and should be) described by viewer-independent representations, whereas the image features and surface representation obtained from images are viewer-dependent. Somehow the two types of representations must be made compatible. Most of the previous systems did this either by quickly hypothesizing the parameters from a very small set of image features or by generating 2-dimensional appearances so that they could work totally in the picture domain. Both were possible because the shapes were fixed and limited.

What is needed is a more systematic way to access the model and to reason about the parameters. The ACRONYM system at Stanford [4] uses invariant and pseudo-invariant features which are predicted from the given object models. In ACRONYM, an object is modeled by its subparts and their affixment (spatial relationships). Volume primitives to represent subparts are generalized cones: A generalized cone is a volume swept by moving a cross section along a 3-dimensional curve (called a spine). When a cross section is a round disc and it is moved along a straight line with its size varying linearly, an ordinary cone is generated. Parameters that specify the generalized cones and their affixment are given not only by specific values but also by free variables with which range and mutual constraints can be associated. In this way one can represent generic object models which allow variations in size and shape. The image features that ACRONYM uses are ribbons (two dimensional strips) and ellipses, which can provide an approximation as the projection of the straight spine and the circular cross section, respectively, of the corresponding generalized cylinder.

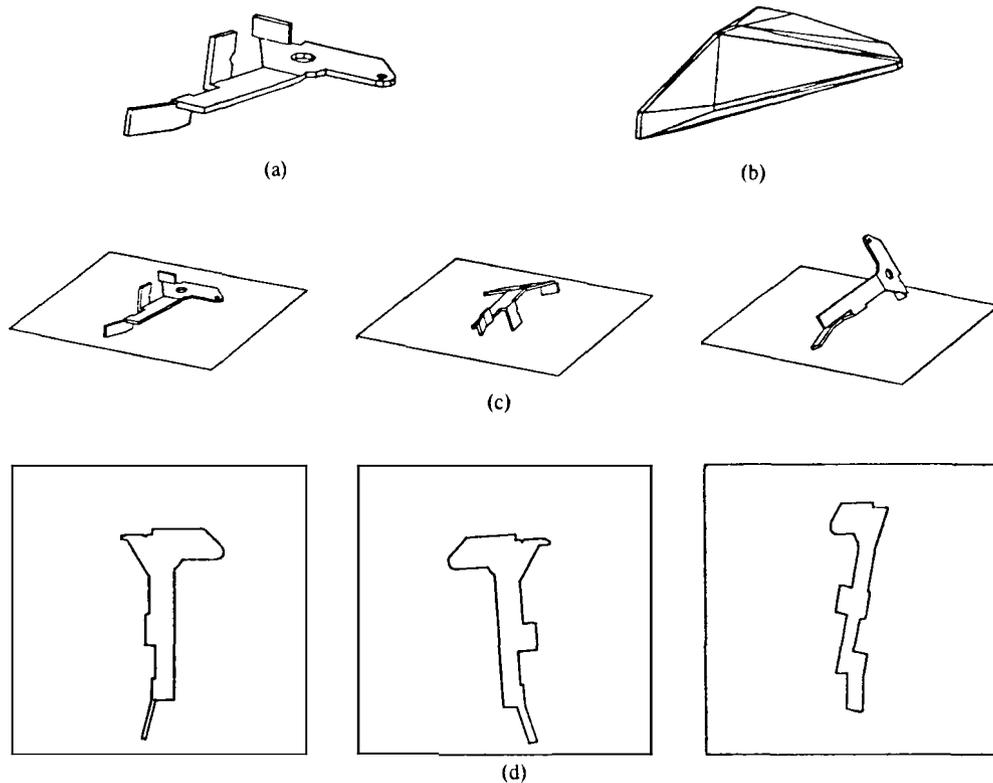


Figure 15: Anticipating the appearances of a part in stable positions. (a) 3D model of a pan; (b) convex hull; (c) computed stable positions; (d) silhouettes of stable postures viewed from overhead.

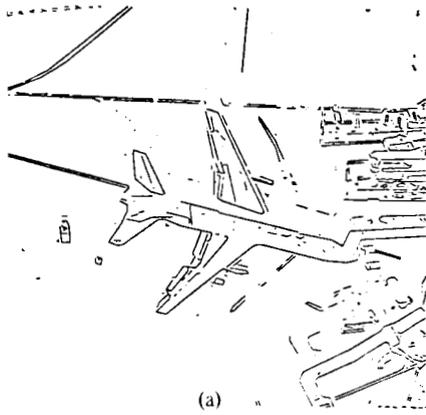
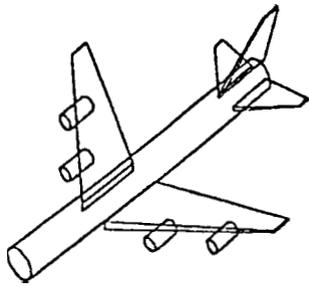
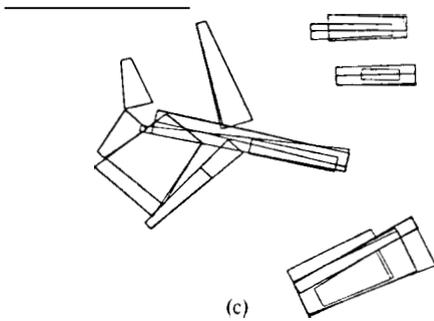
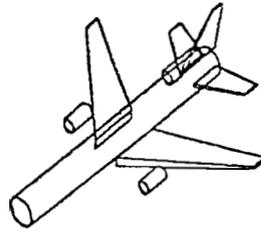


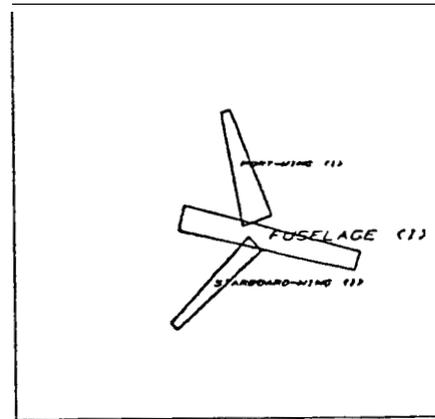
Figure 16: An example of analysis by ACRONYM:
 (a) Line segments for the input image; (b) instances of class models of Boeing-747a and L-1011s; (c) ribbon description; (d) an L-1011 located.



(b)



(c)



(d)

One important idea is to predict from the model invariant and pseudo-invariant features of an object, which will be invariantly observable in the image over the modeled range of variations. They include parallelism, collinearity, and the length ratio and angles between spines. (Here we see an application of the principle of non-accidental regularity.) They provide a coarse filter for hypothesizing possible objects. For hypothesized objects, the unknown parameters (variations of object shape and distance and orientations of the object) are estimated by the algebraic constraints obtained by associating the observation (such as length and angle in the image) with the anticipated range of its value.

Figure 16 shows an example scene: (a) is the line segmented image extracted from input. (b) shows the instances of models of a Boeing-747 and an L-1011; after extracting ribbons (shown in (c)) from the line segment image, an L-1011 was identified as shown in (d). This deduction is not based on the size of the image, but on the relationships between the subparts, such as wingspan-to-fuselage-length.

5. Construction of Scene Description

The goal of the visual process is to construct the description of the environment which is sensed by visual methods: it must be recognized that this is different from classifying parts of an image, detecting the object existence or segmenting the images. The difference is most easily understood in the scenario of robot navigation where classification or detection is not enough to plan actions of the robot: it must have a **3D** description of its environment.

At CMU, Kanade and Herman [16] are developing a system called Incremental **3D** Mosaic, which builds a scene description from a sequence of stereo images. It is currently applied to an urban scene for building a 3-dimensional model of the target area from low-altitude aerial photos. A single pair of stereo photos can give only partial information on the scene: only limited portions are visible in both images and the stereo system cannot be perfect anyway. The information must be incrementally accumulated into a consistent description as new images are available. The information on the portions of the scene which become visible should be added, and the information on the previously known portion should be used to improve the description, either by correcting errors or by increasing the confidence and precision.

Figure 17 shows a typical sequence. We have a stereo pair of images (Figure 17(a)). Lines and junctions are extracted from them (Figures 17(b)). Then, junctions and lines are matched. They are mostly corners and edges of buildings. Here we use the fact that buildings are mostly block-shaped, and they have (gravitational) vertical edges. This allows one to hypothesize the changes of junction appearances along the epipolar line of stereo, and to cope with the difficulty in matching wide-angle stereo images with large disparity jumps, such as urban scenes. We can compute the 3-dimensional locations of the matched junctions and lines, thus forming wireframes. Figure 17(c) shows their perspective view.

The wireframe representation does not yet identify surfaces. The next step is to perform reasoning on surfaces. This is done by assigning planar surfaces so that an enclosed object is obtained with the wireframes as edges. The process is similar to obtaining solid objects from wireframes [21], but we assume that our wireframes are not always complete. Figure 17(d) shows a perspective view of the constructed blocks object. Once we have such a description, we can crop image patches from original images to know the normalized appearances of surfaces (e.g. window patterns). A natural looking display can be generated for the scene viewed from any angle by appropriately transforming such appearances according to surface orientations. Figure 17(e) is an example. Notice that parts of surfaces which were not visible in the original images are displayed distinctively as such (i.e. red color).

This kind of description is useful, for example, for planning the angles that the next images should take: it is generally better to cover as much of the "red" portions as possible to increase the knowledge of the target area. This scenario of 3D Mosaic is applicable to robot navigation and to change detection in a scene.

6. Organization and Control of Vision Systems

A key attribute of an image-understanding system is the interaction between high-level knowledge--object models--and low-level knowledge--image or scene features. While the general flow of information is *bottom-up*, from pixels to image features, to scene features, to object labeling, many systems also have some *top-down* information flow from object models to image features. In the face recognition program by Kanade [13], for example, a model of the arrangement and intensity characteristics of face components guides all of the low-level processing. After finding

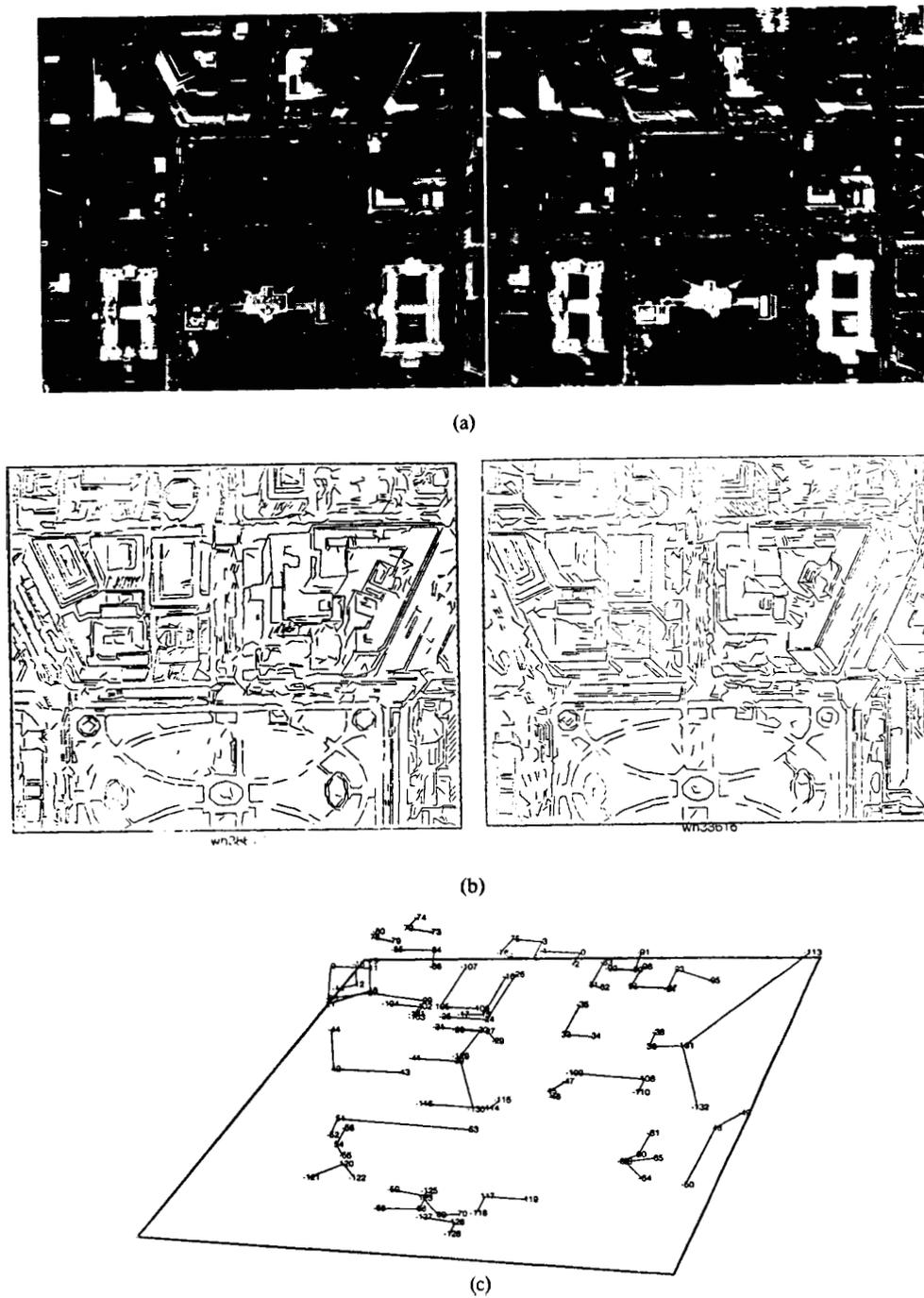


Figure 17: 3D Mosaic: (a) Images of right and left views; (b) line segments extracted; (c) perspective view of 3-D wireframes which are made by edges obtained by stereo matching; (d) plane-surfaced models of buildings; (e) synthesized image of the scene from the angle different from original angle.

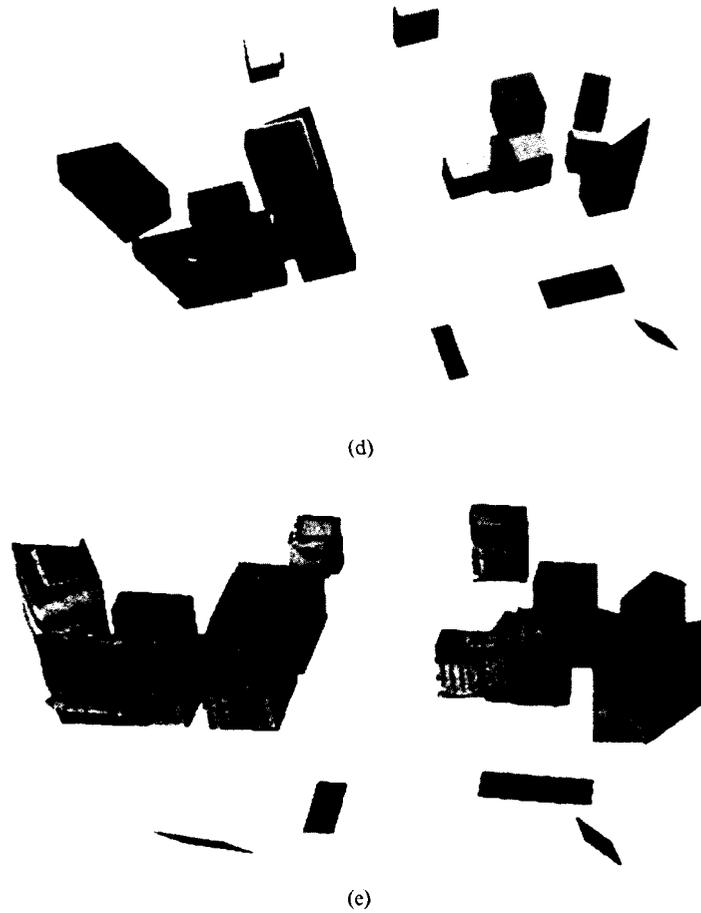


Figure 17: (continued)

the outline of the head, the program estimates the probable position of the eyes, and look for the dark spots that characterize the pupils in and around the predicted locations. When it fails in finding components with such characteristics, it goes back to the previous steps, assuming a certain error there, and try another possibilities. This *model-driven* processing can be both efficient and effective. However, programs that depend very much on high-level control of low-level processing tend to be too domain-dependent and respond poorly when viewing conditions change even slightly. In this section we will examine the organization and control of three different types of vision systems. The following descriptions focus on mechanisms for achieving cooperation and flow of control between low-level and high-level processing stages.

6.1. Production System Organization for Outdoor Scene Analysis

Ohta, Kanade and Sakai [27, 26] developed a semantic region analysis system for outdoor scenes. Given a color image of outdoor scene, the system assigns semantic labels, such as tree and buildings, to regions. As shown in Figure 18 the system consists of bottom-up part and top-down part. Initially the bottom-up part first segments the image into

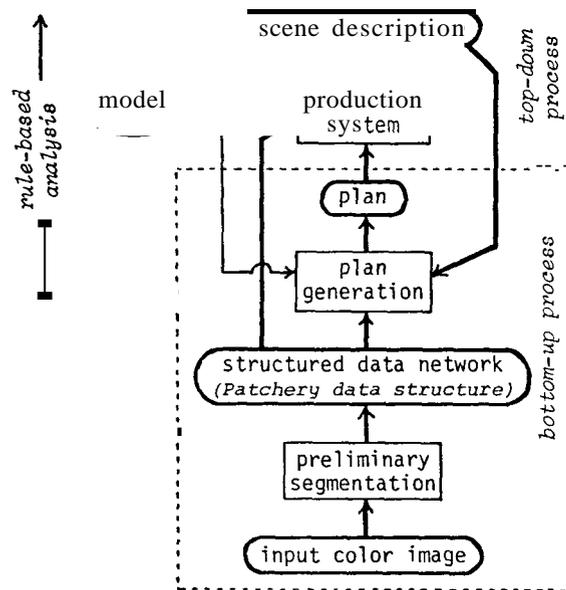


Figure 18: Organization of the region-based scene analysis program by Ohta, Kanade and Sakai

homogeneous regions by an Ohlander-type region segmentation method [28]. The segmentation result is described as a patchery data network which stores properties of all the patches extracted.

Then a *plan* is generated from it by selecting regions by large area, called keypatches. After merging small patches adjacent to keypatches, the system tentatively assigns a set of object labels with corresponding estimates of correctness which are computed first from the unary properties of the keypatches, such as size, shape, and color, and then from the binary relations between them, such as relative positions.

Figure 19 shows an example scene: (a) is an input color image, (b) is the result of preliminary segmentation, and (c) is the plan image. The first plan obtained by using only the unary property rules is shown in Figure 20(a), and Figure 20(b) shows a revised one by using the binary relations.

The top-down process then starts symbolic interpretation of the image by analyzing detailed structure of the scene in the context given by the plan. (The system still can change the interpretations in the plan, in which case the bottom-up process is re-activated.) The analysis uses a production system organization with knowledge of the world (outdoor scene) represented as a set of rules. Each production rule has a condition part and an action part. The condition part is made of fuzzy predicates on properties of and relations between regions. The action part is a set of actions to manipulate the database (patchery network, plan, and scene description) to build the scene description. Each action is described as a form in Lisp. There are TO-DO and IF-DONE rules, corresponding to the antecedent and consequent theorems of PLANNER.

The world model is described as a network of *knowledge blocks (KB)* which define the objects, materials, and concepts in the given task world. The production rules are divided into subsets according to the roles they play in the analysis: For example, the subset for the scene phase analysis is stored in the KB *SCENE* and the subset to analyze the "sky" is stored in the KB *SKY*.

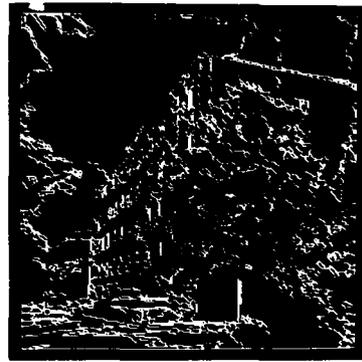
Control of the production system is handled by an agenda which registers all the executable rules (i.e., whose condition parts are satisfied). The analysis iterates the following three steps:

1. An executable action with the highest Score on the agenda is executed. A patch or a set of patches is interpreted and the database is modified.
2. If a keypatch is interpreted in step 1, the control enters into the Scene phase. The production rules included in the KB *SCENE* are activated to (re-)examine the keypatches not yet interpreted. The scene phase, in general, considers the overall structure of the Scene, such as the location of horizon and relationships among objects.
3. Otherwise the control enters into the object phase. Corresponding to the object as which the patch has been just interpreted in step 1. The production rules in the corresponding KB are activated. This phase mainly analyze the local structure related to the particular object.

Figure 21 shows how the plan is modified as the analysis proceeds: (a) when the horizon is determined, and (b) when the outline of the building is extracted. Figure 21(c) shows the final labeled interpretation.



(a) digitized input scene



(b) result of preliminary segmentation



(c) plan image

Figure 19: An outdoor scene:

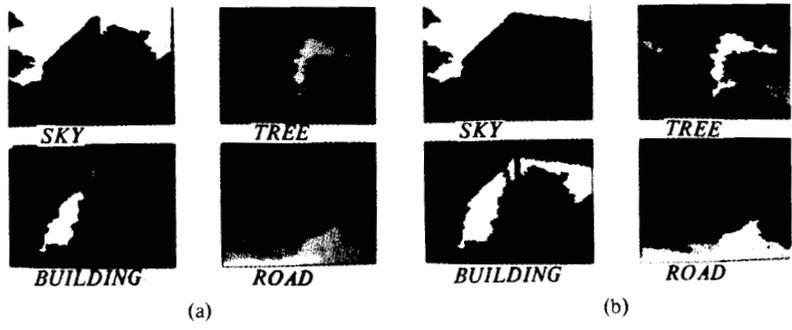


Figure 20: Plan The brightness shows the correctness of the assignments: (a) by use of unary properties of regions; (b) after using binary relations

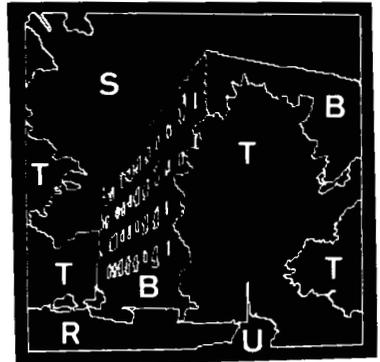
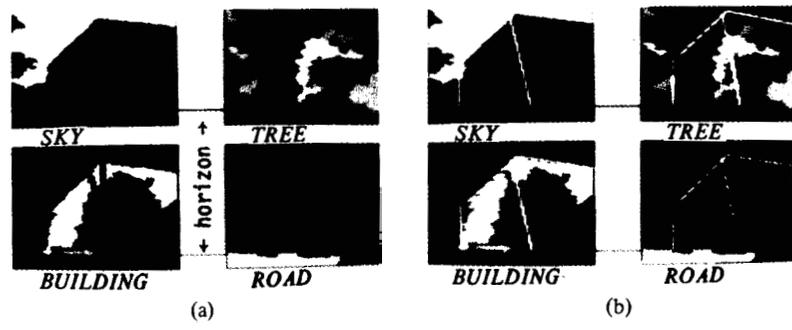


Figure 21: (a) Plan when the horizon is determined. (b) Plan when the outline of the building is determined. (c) final result of interpretation: S: SKY; T: TREE; R: ROAD; B: BUILDING; U: UNKNOWN

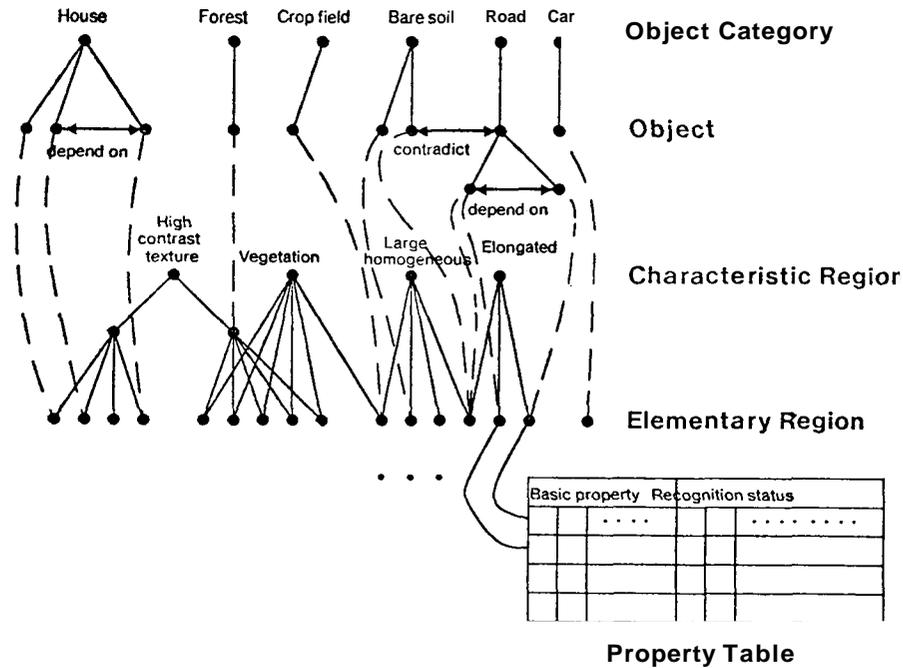


Figure 22: Structure of a blackboard of the system by Nagao, Matsuyama and Ikeda

6.2. Blackboard Architecture for Interpreting Multiband Aerial Photographs

Nagao, Matsuyama, and Ikeda [24, 25] developed a system that interpreted a class of multiband aerial photos. Their image-interpretation system employs multiple, independent knowledge sources that operate on a common, multilevel database. This database, or *blackboard*, is represented as shown in Figure 22. The abstraction levels of image information are *elementary region*, *cue region*, *object*, and *object category*. Models are described in terms of two-dimensional features that can be observed in images. In general, it is not possible to do scene interpretation with two-dimensional models, but it is an acceptable technique for aerial photography because the view angle is so constrained that object shapes change little and occlusion is not much of a problem.

The first step of processing is to smooth the image. A nonsemantic segmenter defines a set of elementary regions—a set of patches that are homogeneous in multispectral properties.

The next step is to extract cue regions. The types of cue regions are large homogeneous regions, shadow and shadow-causing regions, elongated regions, vegetation regions, high-contrast regions, and high-contrast vegetation regions. Each type of cue region triggers one or more object recognizers. Different cue regions may overlap; for example, high-contrast vegetation regions are simply the intersection of high-contrast regions and vegetation regions.

Cue regions are extracted by screening elementary regions: for example, any patch with very low intensity, particularly in red and infrared, is classified as a shadow. An adjacent region with an appropriate boundary on the sunward side is a shadow-maker. Vegetation regions have a high ratio of infrared to red; high-contrast areas are aggregations of small elementary regions. Shadow-making regions trigger the house detector, while high-contrast vegetation regions are likely to be considered forest.

Each elementary region is represented by a node in the lowest level of the blackboard. Nodes at higher levels represent cue regions and objects; they are linked to the elementary regions they subsume. Furthermore, a node can

have a dependency link to another node, indicating that its interpretation was aided by the prior interpretation of the other node.

The property table shown in Figure 22 stores the coordinate range, or *bounding rectangle*, of a region and records whether the region is *unanalyzed*, *recognized*, *irregularly shaped*, or *it rejected*. Each region has only one entry, which means that there can be only one object hypothesis for a region. The first interpretation of a region is kept until a contradiction arises. To resolve contradictions, the system deletes the conflicting region interpretation for which it is least confident. It marks the region as *unanalyzed*, restarting the interpretation of the region; object hypotheses that depend on the deleted node are themselves deleted.

6.3. Schema-based System: VISIONS

The University of Massachusetts' VISIONS system[8] is patterned after the HEARSAY-II speech-understanding system[5]. In VISIONS, hypotheses are posted and accessed on a *blackboard* by independent procedural *knowledge sources*: *KS*. Their activation and scheduling are under the control of a central executive. The system has been tested with outdoor scenes. Figure 23 outlines the structure of VISIONS.

The blackboard in this system represents a layered description of the contents of an image. The lowest levels represent *regions*, *segments*, and *vertices*; they form a structure called an *RSV graph*.

Preprocessing stages are shown in the left half of Figure 23. There are three stages of information representation. The first is the image itself, represented by a resolution pyramid (processing cone). The second stage comprises separate edge and region analysis. The third stage is a merged representation of the results of a correlation between the edge analysis and region analysis. The representations at these *low* levels are of image characteristics, rather than of scene characteristics.

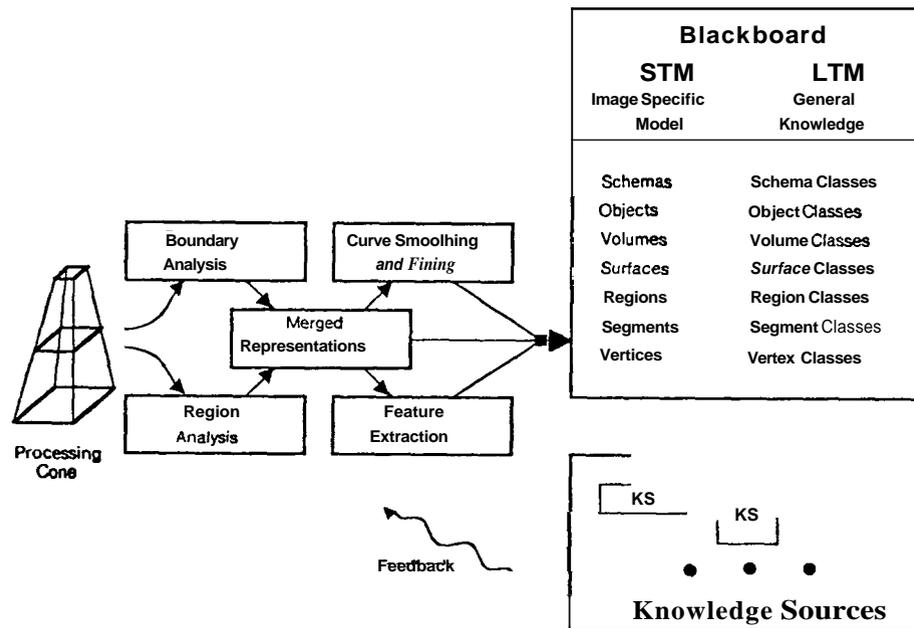


Figure 23: Overview of VISION system by Hanson and Riseman

The next two levels in the blackboard work with surfaces and volumes. At these levels, the system attempts to reconstruct the three-dimensional configuration of the scene. The top two levels work with representations of objects and schemas. At the object level, hypotheses are formed about what the objects in the scene must have been to result in the observed image. The schema level imposes constraints on the selection of object models. There may be office schemas, airport schemas, and so on. Schemas serve the same purpose as Minsky's frames [23].

The blackboard model in Figure 23 illustrates the distinction made in VISIONS between a priori models and image-specific models, though both may be represented in the same manner. The a priori models are stored in *long-term memory* (LTM), while the image-specific models are stored in *short-term memory* (STM). Recognizing that they did not have adequate KSs to make surface and volume hypotheses reliably, the designers of VISIONS compensated by relying heavily on top-down hypotheses represented by models in LTM. By projecting these models into two dimensions, they construct RSV-level models of objects, and these are matched to the actual image.

VISIONS chooses a KS by traversing a decision tree. Its *model builder* decides to expand or to develop a new hypothesis for a model. To expand a model, the *level focuser* first decides which level of the blackboard to work on. Then, that level is expanded under the control of the *node focuser*, the *node expander*, and the *node verifier*. The focuser selects a node from the blackboard to process further, the expander calls a KS to create new hypotheses, and the verifier checks the results for satisfaction of constraints.

6.4. Remarks

Some other interesting computer vision systems that interpret natural scenes include: Shirai [33] (procedural representation), Rubin [29] (constraint network representation) and Ballard, Brown and Feldman [2] (query-oriented analysis). The image/map database MAPS, being developed at CMU by McClellan [22], attempts to use multi-level multi-source knowledge (such as terrain map and cultural maps) for photo interpretation.

7. Conclusion

We have first discussed a general structure of vision, and then identified important areas to work on for realizing a more capable and general vision system than current robotics vision. Emphasis has been put on:

- Computational aspects of vision
- Obtaining constraints from physical and semantic knowledge
- Multi-level representations
- Use of generic models
- Scene descriptions
- Control and information flow in the vision system

References

1. Artificial Intelligence. *Special Issue on Vision* North-Holland, 1981.
2. Ballard, D. H., Brown, C. M. and Feldman, J. A. An Approach to Knowledge-Directed Image Analysis. Computer Vision Systems, New York, 1978. pp. 271-281.
3. Barrow, H. G. and Tenenbaum, J. M. Recovering Intrinsic Scene Characteristics from Images. Computer Vision Systems, New York, 1978, pp. 3-26.
4. Brooks, R. A. "Symbolic Reasoning among 3-D Models and 2-D Images." *Artificial Intelligence* 17(1981), 285-349.

5. Erman, L. D., Hayes-Roth, F., Lesser, V. R., and Reddy, D. R. "The HEARSAY-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty." *Computing Surveys* 12.2 (1980).
6. Falk, G. "Interpretation of Imperfect Line Data as a Three-Dimensional Scene." *Artificial Intelligence* 3(1972), 101-144.
7. Fischler, M. A. and Witkin, A. P. Recovering Intrinsic Scene characteristics from Images. Tech. Rept. Interim Technical Report, SKI International, Nov., 1981.
8. Hanson, A. R. and Riseman, E. M. Segmentation of Natural Scenes. Computer Vision Systems. New York, 1978, pp. 129-163.
9. Horn, B. K. P. "Understanding Image Intensities." *Artificial Intelligence* 8(1977), 201-231.
10. Horn, B. K. P. and Sjoberg, R. W. "Calculating the Reflectance Map." *Applied Optics* 18(1979), 1770-1779.
11. Ikcuchi, K. Shape from Regular Patterns (An Example of Constraint Propagation in Vision). Tech. Rept. A.I. Memo 567, Massachusetts Institute of Technology, March, 1980. Artificial Intelligence Laboratory
12. Ikcuchi, K. and Horn, B. K. P. "Numerical Shape from Shading and Occluding Boundaries." *Artificial Intelligence* 17(1981), 141-185.
13. Kanade, T. *Interdisciplinary Systems Research Volume 41: Computer Recognition of Human Faces*. Birkhauser Verlag, New York, 1977.
14. Kanade, T. "Region Segmentation: Signal vs Semantics." *Computer Graphics and Image Processing* 13(1980), 279-297.
15. Kanade, T. "Recovery of the Three-Dimensional Shape of an Object from a Single View." *Artificial Intelligence* 17(1981), 409-460.
16. Kanade, T. and Herman, M. Incremental 3D Mosaic System. Tech. Rept. (in preparation), Carnegie-Mellon University Computer Science Department, 1982.
17. Kanade, T. and Kender, J. R. Mapping Image Properties into Shape Constraints: Skewed Symmetry, Affine Transformable Patterns and the Shape-from-Texture Paradigm. Tech. Rept. CMU-CS-80-133, Carnegie-Mellon University Computer Science Department, July, 1980.
18. Kender, J. R. *Shape from Texture*. Ph.D. Th., Carnegie-Mellon University Computer Science Department, 1980.
19. Lowe, D. G. and Binford, T. O. The Interpretation of Three-Dimensional Structure from Image Curves. Proc. 7th International Joint Conference on Artificial Intelligence, 1981, pp. 613-624.
20. Mackworth, A. K. "Interpreting Pictures of Polyhedral Scenes." *Artificial Intelligence* 4(1973), 121-137.
21. Markowsky, G., Wesley, M.A. Fleshing Out Wire Frames. Tech. Rept. RC 8124, IBM Research Division, February, 1980. IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598
22. McKeown, D. M. Jr. Maps: Integrating Imagery, Terrain and Map Databases. Proc. IEEE Workshop on Applied Imagery Pattern Recognition, Sept., 1982.
23. Minsky, M. L. A Framework for Representing Knowledge. *The Psychology of Computer Vision*, New York, 1915, pp. 211-277.
24. Nagao, M., Matsuyama, T. and Ikeda, Y. Region Extraction and Shape Analysis of Aerial Photographs. Proc. 4th International Joint Conference on Pattern Recognition, 1978, pp. 620-628.
25. Nagao, M., Matsuyama, T. and Ikeda, Y. Structural Analysis of Complex Aerial Photographs. Proc. 6th International Joint Conference on Artificial Intelligence, 1979, pp. 610-616.

26. Ohta, Y. *A Region-Oriented Image-Analysis System by Computer*. Ph.D. Th., Kyoto University, March 1980.
27. Ohta, Y., Kanade, T. and Sakai, T. *A Production System for Region Analysis*. Proc. 6th International Joint Conference on Artificial Intelligence. Aug, 1979, pp. 684-686.
28. Ohta, Y., Kanade, T. and Sakai, T. "Color Information for Region Segmentation." *Computer Graphics and Image Processing* **13** (Sept. 1980). 222-241.
29. Rubin, S. M. "Natural Scene Recognition Using LOCUS Search." *Computer Graphics and Image Processing* **13** (1980), 298-333.
30. Shafer, S. and Kanade, T. Goniophotometry and the Reflectance Atlas. Tech. Rept. (in preparation), Carnegie-Mellon University, 1982.
31. Shafer, S. and Kanade, T. Using Shadows in Finding Surface Orientations. Tech. Rept CMU-CS-82-100, Carnegie-Mellon University. Jan., 1982.
32. Shafer, S., Kanade, T. and Kender, J. Gradient Space Under Orthography and Perspective. Tech. Rept CMU-CS-82-123, Carnegie-Mellon University, May, 1982.
33. Shirai, Y. Recognition of Man-Made Objects Using Edge **Cuts**. Computer Vision Systems, Sept., 1978.
34. Wesley, M.A., Lozano-Perez, T., Lieberman, L.J., Levin, M.A., Grossman, D.D. "A Geometric Modeling System for Automated Mechanical Assembly." *IBM J. Res Develop.* **24**, 1 (January 1980). **64-74**.
35. Woodham, R. J. "Analyzing Images of Curved Surfaces." *Artificial Intelligence* **17** (1981), 117-141.

