T-PAMI/9/3/13723

New Methods for Matching 3-D Objects with Single Perspective Views

R. Horaud

Reprinted from: IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE,
Vol. PAMI-9, No. 3, MAY 1987.

New Methods for Matching 3-D Objects with Single Perspective Views

RADU HORAUD

Abstract—**In this paper we analyze the ability of a computer vision system to derive properties of the three-dimensional (3-D) physical world from viewing two-dimensional (2-D) images. We present a new approach which consists of a model-based interpretation of a single perspective image. Image linear features and linear feature sets are *backprojected* onto the 3-D space and *geometric models* are then used for selecting possible solutions. The paper treats two situations: 1) interpretation of scenes resulting from a simple geometric structure (orthogonality) in which case we seek to determine the orientation of this structure relatively to the viewer (three rotations) and 2) recognition of moderately complex objects whose shapes (geometrical and topological properties) are provided in advance. The recognition technique is limited to objects containing, among others, straight edges and planar faces. In the first case the computation can be carried out by a parallel algorithm which selects the solution that has received the largest number of votes (accumulation space). In the second case an object is uniquely assigned to a set of image features through a search strategy. As a by-product, the spatial position and orientation (six degrees of freedom) of each recognized object is determined as well. The method is valid over a wide range of perspective images and it does not require perfect low-level image segmentation. It has been successfully implemented for recognizing a class of industrial parts.**

Index Terms—**Geometric constraints, inverse perspective transform, spatial reasoning, three-dimensional object recognition.**

I. Introduction

ONE of the fundamental characteristics of the human vision system is its ability to derive properties of the three-dimensional (3-D) physical world from viewing two-dimensional (2-D) images. There are two major phenomena involved in this process which make it difficult to understand both from a psychophysical and computational point of view. The first is the ambiguity of the image data due to the fact that spatial information is lost by projection. The second is the presence in the image of random processes inherent to complex scenes. Imagine a scene such as a countryside landscape. How do we perceive the relatively simple geometric structure that gives rise to the scene? How do we make abstraction of irrelevant details that do not reflect intrinsic characteristics? How do we recognize the objects composing the scene?

When trying to answer these questions we have two goals in mind: 1) to suggest a sound computational theory for supporting qualitative experiments performed with human subjects and 2) to implement this theory within a computer vision system. Perception is a complex phenomenon. It involves the interpretation of sensory data in terms of physical world entities and it uses prior knowledge about the structure of the world. Previous computational approaches have generally concentrated on one particular visual process: stereopsis, motion, and shape from monocular cues are among the most investigated ones for which partial results on the kind and amount of computation needed to be carried out are now available.

A widely prevailing theory (see Marr [1]) postulates that these more or less autonomous processes produce a description in terms of local surface properties, the *2.5-D representation*, from which instances of objects are found as generalized cylinders, and finally a description of the scene in terms of these objects is produced.

The key idea of our approach is that a single view is in most of the cases sufficient for retrieving an unambiguous three-dimensional interpretation. Nonetheless, although the information is there, it is not straightforward to actually find the correct 3-D solution within a reasonable time. A sequence of images can greatly simplify the computational effort.

Our work is also closely related to recent results in experimental and cognitive psychology. In order to demonstrate that mental processes simulate physical world ones, Cooper and Shepard [2] showed to human subjects pairs of computer-generated line drawings. In one pair, identical structures were displayed with different spatial orientations. The subjects were asked to match the two structures. The results suggested that, first they imagined the structures as 3-D objects (and not as image features!) and second they mentally performed the rotation allowing the "object" from one image to overlap onto the "object" from the other image. This transform was imagined in three-space. A remarkable result was the fact that this *spatial reasoning* was independent whether the two objects differed by a rotation in depth or by a rotation in the image plane. We believe that the same type of reasoning occurs in the presence of more complex sensory data. The basic mechanism for recognizing line drawings should be similar to the one involved by natural scene interpretation.

In the model we propose for understanding these experiments from a quantitative point of view and for inter-

Manuscript received July 19, 1985; revised November 10, 1986. Recommended for acceptance by W. E. L. Grimson. This work was performed while the author was with the Laboratoire d'Eletronique et des Techniques de l'Informatique, Grenoble, France.

The author is with LIFIA, BP 68, 38402 Saint-Martin d'Hères, France.

IEEE Log Number 8613723.

preting single views in general, *there is no 2.5-D sketch*. Our approach is related to recent work in shape from contour (Kanade [3], Barnard [4], [5], Brady and Yuille [6]) and can be summarized as follows. A simple low-level process produces a description of the image in terms of linear features. These features are then *backprojected* onto the 3-D space where they are meant to satisfy several types of constraints. Simple global properties of the physical world such as parallelism and orthogonality [4], [5] are used to derive a scene centered representation. Backprojected features may also satisfy such constraints as uniformity [4], [7], compactness [6], or may be grouped through search processes for identifying objects. Finally a scene centered description may be constructed in terms of the spatial locational parameters of these objects.

In order to recover 3-D shape from 2-D projections, explicit models of both the perspective transform and the physical world are essential. In this paper we shall consider central (perspective) projection exclusively since it is the correct model both for the human eye and for cameras. A general framework for representing the physical world is beyond the scope of this paper. A simple, general purpose scheme for describing the shape embedded in 3-D geometric structures will be used. Although this scheme is valid for generic surface primitives such as planes, cylinders, cones, etc., we will limit our discussion to objects bounded by planar faces. In [4], Barnard devised some methods for computing vanishing points and orientation of planes based on backprojection of angle magnitude and curvature. He used the *Gaussian sphere* for representing geometric constraints. The domain of applicability of his methods is limited to images with strong perspective effects. This restriction is unacceptable since objects in the scene may appear over a wide range of depths. Cameras for machine vision may be equipped with a large variety of lenses. In conclusion, a method applicable to many image formation situations is needed.

The approach that we advocate in this paper can be paraphrased as follows. A simple segmentation process performs line finding and line grouping. This process is mainly data-driven. However, it takes into account descriptive properties that are invariant under perspective. On this premise lines are grouped into angles (coincidence of two lines), junctions (three lines) or more. These image attributes are backprojected onto the 3-D space using the geometric constraints of perspective. These constraints are combined with specific knowledge about the class of possible spatial shapes in order to infer metric properties such as orientation and depth. The method is then applied to compute the viewer-to-world relation using the orthogonality assumption. A more general object recognition procedure is next described. The backprojected image features are put into correspondence with object features. This matching is carried out by a hypothesize-and-test strategy implemented as a depth-first tree search. Finally we apply the method to the task of finding the orientation (three rotations) and position (three translations) of an industrial part.

II. Backprojection of Image Lines

In this section we shall consider the perspective camera model and descriptive and metric properties for deriving spatial constraints on the backprojection of image lines. When interpreting image attributes as spatial figures we seek solutions that are valid when the scene is being viewed from a general position so that the perceived configuration is not an accident due to a particular viewpoint. Under this assumption, lines intersecting in the image are interpreted as lines intersecting in space. In order to derive spatial orientation constraints we shall make assumptions about the values of the angles that these lines make in space.

A. The Geometry of Perspective

The viewer (sensor, camera) centered coordinate system we shall use throughout the paper has its origin at the focal point F. The image plane is parallel to the xy-plane at distance f (the focal length) from the origin along the z-axis (see Fig. 1). A space point projects onto the image along a line passing through the focal point. This is the perspective transform. If the coordinates of the space point are x, y, and z, the coordinates of its projection are xf/z, yf/z, and f. Let us now associate unit vectors either with directions of lines or with normals to planes. A unit vector may be represented as a point on a unit sphere centered at the origin, the Gaussian sphere. A point on this sphere has two angles as coordinates, the azimuth α and the elevation β. The azimuth is the angle measured from the z-axis in the yz-plane. The elevation is the angle measured from the yz-plane toward the x-axis (Fig. 2). The relationship between the Cartesian coordinates and the spherical coordinates (azimuth and elevation) of a point lying on the unit sphere is:

$$\begin{aligned} x &= \sin \beta \\ y &= \sin \alpha \cos \beta \\ z &= \cos \alpha \cos \beta \end{aligned} \tag{1}$$

In practice only half of the sphere (the one oriented toward the viewer) is of importance. This surface can be represented digitally as a 2-D grid with α, horizontal varying from $\pi/2$ to $3\pi/2$ and β, vertical varying from $-\pi/2$ to $\pi/2$. Geometric constraints may be represented as loci of points (curves) on this surface. Since perspective is a nonlinear transform we shall use *constructive* rather than analytical methods to determine intersections of curves, i.e., solutions, on the sphere. To summarize the orientation of any plane (including the image plane) and the direction of any line (including image lines) will be represented as points on the Gaussian sphere. Equations derived both from the camera geometry and from metric properties will constrain these points to lie on certain curves.

B. Backprojection of Lines

Let us now associate an *interpretation plane* with an image line. This plane passes through the focal point and

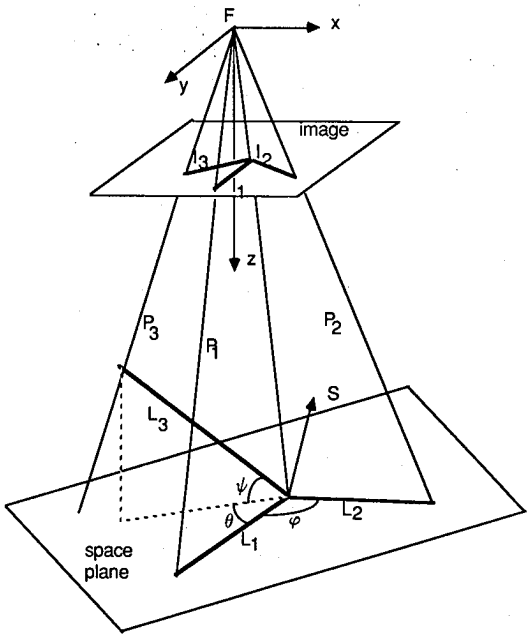

Fig. 1. The geometry of the perspective transform. An interpretation plane is associated with an image line. The shape of a three-line vertex (L_1, L_2, L_3) is defined by three angles, ϕ, θ, and ψ. Its space orientation is constrained by its image projection (l_1, l_2, l_3).

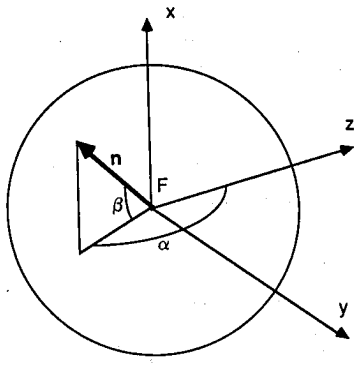

Fig. 2. A unit space vector n may be represented as a point on the surface of the Gaussian sphere.

contains the image line. Therefore its equation can be easily computed from this line. Since it passes through the origin of the camera frame, it intersects the Gaussian sphere along a *great circle* which is the locus of possible directions of the lines belonging to the interpretation plane. Hence, an image line backprojects as a great circle on the Gaussian sphere. If we denote by $\boldsymbol{P}$ the vector normal to the interpretation plane with Cartesian coordinates P_x, P_y, and P_z and by $\boldsymbol{L}$ the direction vector of a line belonging to this plane, the equation of the great circle is (the tip of $\boldsymbol{L}$):

$$\boldsymbol{P} \cdot \boldsymbol{L} = 0. \tag{2}$$

Or, in spherical coordinates:

$$P_x \sin \beta + P_y \sin \alpha \cos \beta + P_z \cos \alpha \cos \beta = 0 \tag{3}$$

which is the locus (on the Gaussian sphere) of the spatial interpretations of an image line.

C. Backprojection of Angles

Let us consider now two image lines intersecting in an image point. We shall call this figure an *image angle*. We seek a spatial orientation constraint for the space plane containing the *space angle* which produced, by projection, this image angle, i.e., [4]. We denote by l_1 and l_2 the two image lines, by $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ the normals to their associated interpretation planes, and by $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ the spatial backprojections of l_1 and l_2: $\boldsymbol{L}_1$ belongs to the interpretation plane of l_1 and $\boldsymbol{L}_2$ belongs to the interpretation plane of l_2. Moreover, $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ are constrained to be coplanar and to form a space angle whose value is φ. We denote by $\boldsymbol{S}$ the normal to the plane formed by $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$. $\boldsymbol{L}_1$ is the intersection of the interpretation plane $\boldsymbol{P}_1$ and the space plane $\boldsymbol{S}$; hence it is the cross-product of the normals to these planes (Fig. 1):

$$\boldsymbol{L}_1 = \boldsymbol{S} \times \boldsymbol{P}_1 \tag{4}$$

$$\boldsymbol{L}_2 = \boldsymbol{S} \times \boldsymbol{P}_2. \tag{5}$$

The dot-product of $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ is the product of their magnitude times the cosine of the angle between them:

$$\boldsymbol{L}_1 \cdot \boldsymbol{L}_2 = \| \boldsymbol{L}_1 \| \, \| \boldsymbol{L}_2 \| \cos \varphi. \tag{6}$$

Using the heuristic that the value of φ is imposed and by combining (4), (5), and (6), we can derive a constraint on the orientation of the space plane:

$$(\boldsymbol{S} \times \boldsymbol{P}_1) \cdot (\boldsymbol{S} \times \boldsymbol{P}_2) = \| \boldsymbol{S} \times \boldsymbol{P}_1 \| \, \| \boldsymbol{S} \times \boldsymbol{P}_2 \| \cos \varphi. \tag{7}$$

D. Backprojection of Junctions

We shall consider now three image lines intersecting in an image point, l_1, l_2, and l_3. We call this figure an *image junction*. We seek spatial constraints for the orientation of the figure formed by $\boldsymbol{L}_1$, $\boldsymbol{L}_2$, and $\boldsymbol{L}_3$ which are the backprojections of l_1, l_2, and l_3. If an image junction is not an accident of viewing $\boldsymbol{L}_1$, $\boldsymbol{L}_2$, and $\boldsymbol{L}_3$ from a particular viewpoint, these lines form a *space junction*. Let $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ form a space angle just as above which provides a constraint for the orientation of the plane $\boldsymbol{L}_1 - \boldsymbol{L}_2$ of the form of (7).

Let us consider two additional angles, θ and ψ defining the position of the third line $\boldsymbol{L}_3$ relatively with $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$. θ is the angle made by $\boldsymbol{L}_1$ and the projection of $\boldsymbol{L}_3$ onto the space plane $\boldsymbol{S}$. ψ is the angle made by $\boldsymbol{L}_3$ and this projection (Fig. 1). Note that the set of three angles φ, θ, and ψ constrain the figure formed by $\boldsymbol{L}_1$, $\boldsymbol{L}_2$, and $\boldsymbol{L}_3$ to be rigid. Let us consider a local frame defined by $\boldsymbol{S}$ as the x-axis, $\boldsymbol{L}_1$ as the y-axis, and their cross-product as the z-axis. $\boldsymbol{L}_3$ can be written as:

$$\boldsymbol{L}_3 = \boldsymbol{S} \sin \psi + \boldsymbol{L}_1 \cos \theta \cos \psi + (\boldsymbol{S} \times \boldsymbol{L}_1) \sin \theta \cos \psi. \tag{8}$$

Since $\boldsymbol{L}_3$ belongs to the interpretation plane associated with l_3, the dot-product of $\boldsymbol{L}_3$ and $\boldsymbol{P}_3$ is zero:

$$\boldsymbol{L}_3 \cdot \boldsymbol{P}_3 = 0. \tag{9}$$

By combining (4), (8), and (9), one obtains the following additional constraint for the orientation of the space plane S (see the Appendix for details concerning the derivation of this equation):

$$(S \cdot P_3) \| S \times P_1 \| \sin \psi + S \cdot (P_1 \times P_3) \cos \theta \cos \psi + (S \cdot P_1)(S \cdot P_3) \sin \theta \cos \psi = (P_1 \cdot P_3) \sin \theta \cos \psi. \quad (10)$$

Equations (7) and (10) express the constraints on the orientation of the space plane S derived from the geometry of perspective and from the rigidity assumption concerning the space figure formed by L_1, L_2, and L_3. These two equations contain five unknowns, that is the spherical coordinates of S, α, and β and the angles defining the shape of the space figure, φ, θ, and ψ. In order to obtain explicit solutions for the orientation of the space plane one has to specify the space angles (φ, θ, and ψ) and to solve analytically the set of equations (7) and (10). However, since these equations are nonlinear, such an explicit solution may be difficult to obtain. Instead we shall use a constructive method. Equations (7) and (10) can be written as:

$$\cos \varphi = f(\alpha, \beta) \quad (11)$$

and

$$\sin \theta \cos \psi = g_1(\alpha, \beta) \sin \psi + g_2(\alpha, \beta) \cos \theta \cos \psi + g_3(\alpha, \beta) \sin \theta \cos \psi. \quad (12)$$

At each possible orientation of S, i.e., for each couple (α, β) over the whole domain (the half of the unit sphere looking toward the viewer) we produce a value for f, g_1, g_2, and g_3. This computation is carried out only once for each image junction, regardless of its spatial interpretation. Whenever a set of specific values for the space angles are available, the curves involved by (11) and (12) are constructed point by point. The intersections of these curves are possible solutions for the orientation of the vector S.

Let us consider an example. Fig. 3(a) shows an image junction. This junction is the result of projecting a true 3-D three-edge vertex onto the image plane. In this experiment the focal length f is of 25 mm and the image size is of 8.8 mm by 6.6 mm. With this setting the perspective effects are moderate.[1] Fig. 3(b)–(d) shows *contours on the Gaussian sphere* (α, horizontal and β, vertical) obtained by combining the backprojection of the junction of Fig. 3(a) with various space angle values. In Fig. 3(b) the junction is being interpreted as a vertex with angles $\varphi = 60°$, $\theta = 120°$, and $\psi = 80°$. In Fig. 3(c) the values of these angles are $\varphi = 110°$, $\theta = 60°$, and $\psi = 45°$, while in Fig. 3(d) all the angles are 90°.

There are either no solutions at all, one solution, or two solutions. In this last case the two solutions correspond to two different spatial orientations of the same space figure: one with the concavity toward the viewer and the other reversed, that is with the convexity toward the viewer. These two solutions are equally likely. This may be the basis of a computational explanation for some perceptual reversals. Notice that there might be other solutions on the hidden side of the sphere.

[1]In all our experiments the focal length varied from 10 to 90 mm.

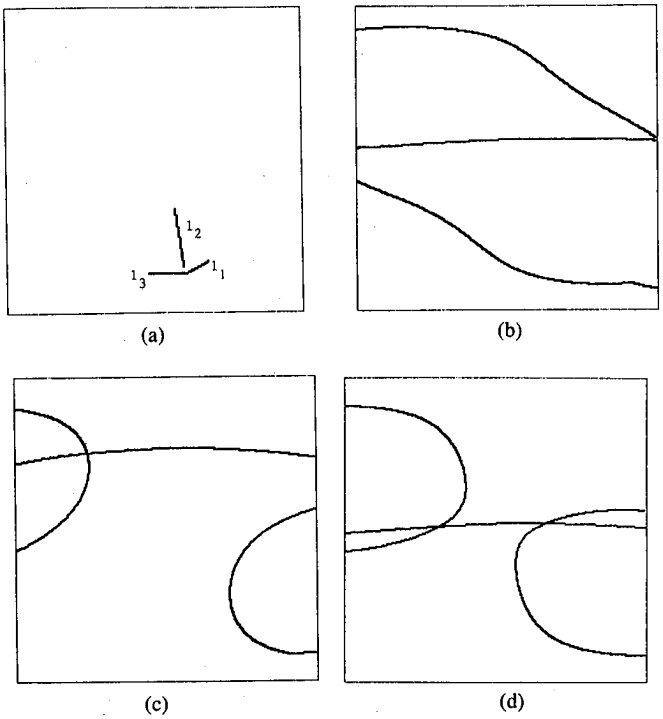

Fig. 3. (a) An image junction. (b) Contours on the Gaussian sphere represented here digitally on a 2-D grid with α horizontal and β vertical. The space angles ϕ, θ, ψ are, respectively, 60°, 120°, and 80°. (c) One solution for the orientation of S (see text) when the space angles are 110°, 60°, and 45°. (d) Two solutions when all the space angles are equal to 90°.

In conclusion, one view is not sufficient for retrieving 3-D properties of the physical world. One needs to make stringent shape assumptions. But even if this is the case, such things as lines and angles still have an infinity of spatial orientations. A three-line junction has a finite set (0 to 2) of spatial orientations. This might be the reason for which line-drawings with junctions are usually interpreted as 3-D shapes whereas line-drawings with angles are not. Moreover, the backprojection of junctions is not dependent on strong perspective effects. This is not the case with the backprojection of image polygons being interpreted as planar figures as is done in [4].

The next sections will explore this backprojection method within the context of two different algorithms which explore prior knowledge in order to produce a description in terms of 3-D objects.

III. Example: Cube Images

In this section we shall use backprojection in conjunction with a global orthogonality assumption. This assumption may be very useful when dealing with man-made environments: the force of gravity induces certain orientations and shapes to be more likely than others (Kender [8]). Vertical and horizontal surfaces are preponder-

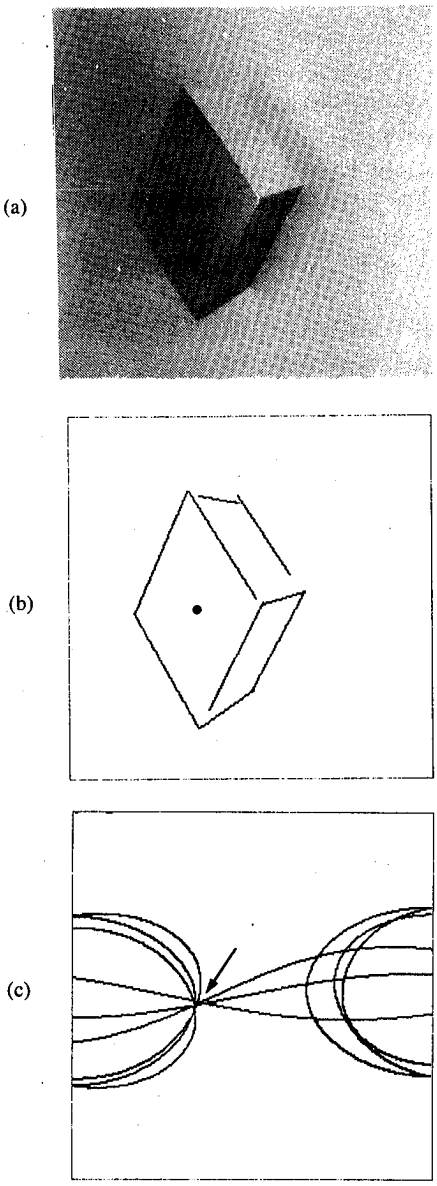

Fig. 4. (a) The gray-level image of a cube taken with a 10 mm lens. (b) The lines extracted and manually selected from this image. (c) The spatial orientation of one face (marked with a dot) obtained by overlapping the contribution of all the junctions touching this face.

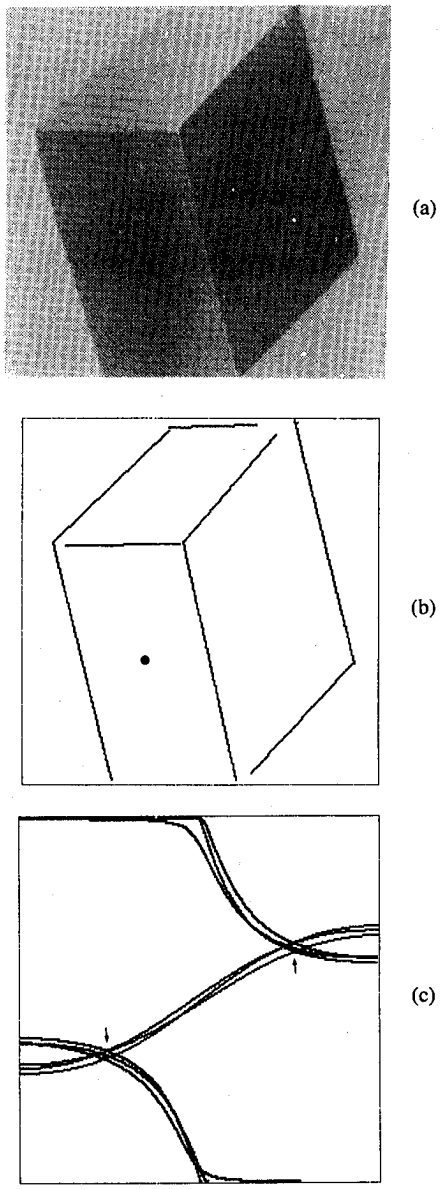

Fig. 5. Same as Fig. 4 where the focal length of the lens is 90 mm. There are two possible solutions for the orientation of the marked face.

ant. Unfortunately, three-space surfaces do not appear directly in an image. Hence, one has to deal with image cues in order to determine vertical and horizontal planes. Image lines are good candidates for deriving these properties.

In order to illustrate the effectiveness of using backprojection from recovering the viewer-to-scene correspondence, we shall use a cube being viewed from an unknown position and orientation.

A line finding algorithm extracts linear edges from the image of a cube. This segmentation process will be described later. Fig. 4(a) shows the cube taken with a television camera through a 10 mm lens. Fig. 5(a) shows the same cube through a 90 mm lens. Figs. 4(b) and 5(b) show the result of line finding where noisy lines have been *manually* eliminated. The remaining lines can be easily grouped to form angles and junctions. A polygon finding algorithm detects four-line faces. This algorithm is described in a companion paper, [9]. We concentrate on the derivation of the spatial orientation of one face of the cube, say the face marked with a dot (•) in Figs. 4 and 5.

Let us backproject the image junctions that share two lines with this face and assuming right vertices in three-space. When all the space angles are equal to 90°, (7) and (10) become:

$$(\boldsymbol{S} \times \boldsymbol{P}_1) \cdot (\boldsymbol{S} \times \boldsymbol{P}_2)/(\| \boldsymbol{S} \times \boldsymbol{P}_1 \| \, \| \boldsymbol{S} \times \boldsymbol{P}_2 \|) = 0 \tag{13}$$

and

$$\boldsymbol{S} \cdot \boldsymbol{P}_3 = 0. \tag{14}$$

The contribution of a junction is recorded as a set of two votes (two spatial orientations) on the Gaussian sphere. Fig. 4(c) shows the result of three image junctions voting for the orientation of the face of the cube. There is a unique solution for the orientation of this face, i.e., the vector S. The same result may have been obtained with any other face. Let us repeat this experiment with the second image which is not subjected to a strong perspective distorsion. In this case, as is shown in Fig. 5(c), there are two solutions for the orientation of each face.

These results are consistent with the *Necker's cube illusion*: the perceptual reversal due to the existence of two spatial solutions disappears under strong perspective effects. It is quite hard to determine with precision when this illusion disappears since the distortion varies continuously with the focal length. Kanade [3] developed an analytical solution in the case of orthographic projection but his method requires the measurement of the skewed symmetry of all the faces forming a junction.

The use of the orthogonality assumption implies the simplest solution from a computational point of view [5]. Experiments made by psychologists with human perception (Gregory [10], Rock [11]) tend to show that people apply abundantly this kind of simple heuristics. Since there is no way to provide confirmation whether the assumption is valid (unless a specific object has been recognized), it may lead to perceptual illusions. There may be a link between the computing technique developed here and the psychophysical theory.

IV. Object Recognition

In this section we shall show how backprojection may be combined with the geometric model of a 3-D object in order to recognize instances of this object in a single intensity image. The task of recognition consists of establishing a unique correspondence between a set of image features and a specific 3-D object model. As a by-product, the locational parameters (three rotations and three translations) of the recognized objects are also determined.

The originality of this method over previous approaches (Goad [12], Brooks [13], Lowe [14]) relies on the fact that the tree-like search process is entirely performed in three-space. In the ACRONYM system developed by Brooks [13], 2-D appearances of 3-D objects are predicted. These predictions are then matched to an image. Goad [12] described a two-stage object recognition technique. The recognition consists of a predict-observe-backproject sequence implemented as a depth-first tree search. An object linear edge is represented as a locus of points on the unit sphere. This locus contains all the directions from which this edge is visible and it is a function of the shape of the object. Hence it can be computed off-line. The basic loop of the algorithm is the following: an object edge is selected and based on its Gaussian sphere locus, its position and orientation in the image is predicted; the list of image edges is checked to see whether any has the predicted qualities; if an edge with the predicted qualities is found, the match is extended to include this edge and its measured image position and orientation are used to refine the current hypothesis. The current hypothesis is a range of possible object-to-camera relationships, i.e., a locus on the Gaussian sphere. This locus is narrowed down any time a new image-line-to-object-edge match is found. This method does not take into account the dramatic reduction in the number of spatial solutions associated with feature grouping.

The method implemented by Lowe [14] performs line-to-line matching. It consists of an iterative method that solves for the viewpoint determination. This technique does not explore the tighter geometric constraints available with the backprojection of linear features.

After outlining the low-level image segmentation process and the object modeling scheme, we concentrate on the image-to-object correspondence algorithm. This algorithm predicts an object through an image-feature-to-object-feature assignment and verifies this hypothesis by checking whether other similar assignments are compatible with the first. This algorithm is similar to maximal-clique finding methods, i.e., Bolles and Cain [15] and Horaud and Bolles [16], and is implemented here as a depth-first tree search.

A. Image Segmentation

The approach that we envisage throughout the paper assumes that a certain number of linear features must be detected in the image. Unfortunately, image segmentation is not a completely well understood process and a short discussion is worthwhile. Consider for example an image line which is a step edge in intensity. This line may represent a geometric feature such as a real object edge or a false tangent edge which is the line of least curvature of a surface curving away from the image; it could be as well a shadow or some noise due to small surface irregularities. In the absence of higher-level knowledge it is impossible to perform line labeling correctly. The advantage of our method is that it does not require perfect segmentation and it performs line labeling as it recognizes objects. We shall describe here a simple, general purpose segmentation scheme. Undoubtedly, the better the segmentation, the faster the recognition.

The segmentation starts with detecting edges as zero-crossings in the lateral inhibited image, i.e., the convolution of the image with the difference-of-Gaussian operator (Marr and Hildreth [17]). Zero-crossings whose slopes are too small are disregarded. Next, a simple edge follower algorithm links these edges into edge-chains using image connectivity properties. Each edge-chain is considered separately and is piecewise approximated with straight line segments. This process uses a classical split-and-merge control structure (Pavlidis and Horowitz [18]). At this stage, short lines are regarded as noisy data and eliminated.

We have implemented a relatively straightforward algorithm for finding image angles and junctions. The lines are first considered pairwise in order to be grouped into

angles: the intersection of a line-pair is first computed. If this intersection falls within the image bounds and if each line has one of its ends closed enough to this intersection, an angle is marked. Angles that share a common intersection and a common line are fused into a junction.

The last data-driven process consists in a polygon finding algorithm. The problem of finding image polygons is equivalent to the problem of finding cycles in a graph. An image junction is mapped into a graph node. An image line is mapped into a graph arc. Two nodes are connected by an arc whenever their corresponding image junctions share a common line. Therefore, finding an image polygon is equivalent to a graph search. The amount of search varies linearly with the number of lines. This is due to the *planarity* of the graph: a graph is said to be planar if it can be drawn on a sheet of paper in such a manner that its arcs never cross each other. This is a direct consequence of the fact that lines never cross in the image.

B. Object Modeling

The function of modeling within the context of matching is to enumerate the geometrical and topological properties of an object and to organize them in an efficient manner. One of our basic assumptions is that recognition should be based on a few features rather than on many (Bolles, Horaud, and Hannah [19]). Using this approach, almost any matching strategy, even an exhaustive search can rapidly determine the best match. The problem is to be able to find those few features.

Standard object representation schemes have been mainly designed for constructing (*constructive solid geometry—CSG*) and displays (*boundary representation—B-rep*) parts. These schemes must be completed for visual recognition tasks and this is one goal of our research. One possibility is to derive a vision-oriented model from these CAD-like models. Hence, our object representation scheme has two parts:

1) The CAD model contains a standard volume-surface-edge-vertex description as well as a network of pointers linking topologically connected features.

2) The vision-oriented model is a redundant representation: a feature is explicitly listed under several classifications and is evaluated according to its expected utility.

For example, a linear edge appears in the list of edges where all its parameters are listed explicitly, appears as well in the list of edges bounding a planar face, in the list of vertices, etc. It may also appear in such lists as parallel or/and collinear edges.

The main criteria for determining the utility of a feature are its model type (convex or concave, linear or circular, symmetric or asymmetric, short or long), the frequency with which it is encountered in the object and the likelihood of it being detected in the data. These criteria allow the ranking of object features into best features, second best features, and so forth. This ranking is very useful since as with all tree searches it is important to order in advance the alternatives according to their expected utility. We have implemented a planning strategy that produces for each object a customized representation, i.e., it automatically establishes sets of features that should be combined at runtime for efficient recognition. The ranking and selection of features varies from one object to another. A line is intrinsically more ambiguous than a circular arc. However, within the context of a specific object a linear edge may be more useful for recognition if there are only a few such edges in the object.

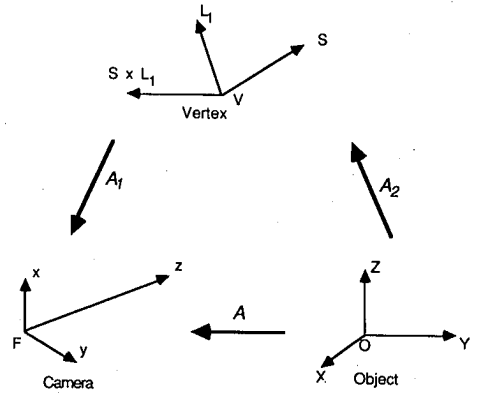

Fig. 6. The object-to-camera transform A may be decomposed into an object-to-vertex transform A_2 followed by a vertex-to-camera transform A_1.

C. Image-to-Object Correspondence

There are two processes within the matching algorithm. The first process consists of computing the locational parameters of an object in sensor coordinates given an image-feature-to-object-feature assignment. Such an assignment may provide either bounds or exact values for the locational parameters. The second process consists of finding a set of mutually compatible assignments that uniquely determines the locational parameters of the object. These parameters are the three rotations and three translations embedded in the transform that maps an object centered frame into the camera frame. We shall represent this transform by a four by four matrix A (standard homogeneous coordinates). There are nine coefficients that specify the three rotations and three coefficients that specify the three translations. This transform may be decomposed into a vertex-to-camera transform A_1 followed by an object-to-vertex transform A_2 (see Fig. 6):

$$A = A_1 A_2. \tag{15}$$

Let us show first how the coefficients of these matrices are related to the geometry of perspective and to the model of the object being observed. Matrix A_1 is of the form:

$$A_1 = \begin{pmatrix} p_x & q_x & r_x & t_x \\ p_y & q_y & r_y & t_y \\ p_z & q_z & r_z & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{16}$$

Consider a set of three image lines l_1, l_2, and l_3 forming a junction and let L_1, L_2, and L_3 be their object assignments: L_1, L_2, and L_3 belong to an object vertex. Con-

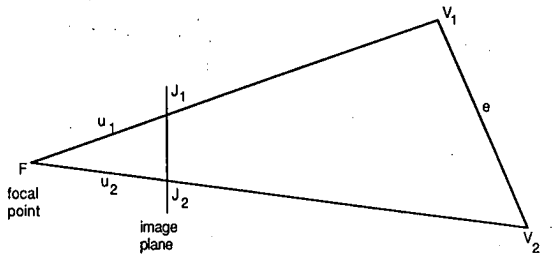

Fig. 7. The computation of depth from two junction-to-vertex assignments.

sider a vertex-centered frame whose origin is the tip of this vertex V: the x-axis is the direction normal to the plane formed by $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$, i.e., $\boldsymbol{S}$, the y-axis is the direction of $\boldsymbol{L}_1$, and the z-axis is the direction of the cross-product of $\boldsymbol{S}$ and $\boldsymbol{L}_1$. Since the shape of this vertex is determined by the model, one can compute the direction of $\boldsymbol{S}$ in camera coordinates through equations (7) and (10) and the direction $\boldsymbol{L}_1$ through (4). Hence, the rotational part of the vertex-to-camera transform A_1 is completely defined by these three vectors: p_x, p_y, and p_z are the coordinates of $\boldsymbol{S}$ in the camera frame, q_x, q_y, and q_z are the coordinates of $\boldsymbol{L}_1$ and r_x, r_y, and r_z are the coordinates of the cross-product of $\boldsymbol{S}$ and $\boldsymbol{L}_1$.

Let us now compute the translational parameters, that is the camera coordinates of the vector $\boldsymbol{t}$ from the focal point F to V. Let d be the distance from F to V: the tip of the junction J is the projection of V onto the image plane. Its camera coordinates are $(J_x, J_y, f, 1)$. Let $\boldsymbol{u}$ be the vector from F to J. See Fig. 7. The camera coordinates of the vector $\boldsymbol{u}$ are $(J_x, J_y, f, 0)$. The vector $\boldsymbol{t}$ is therefore given by:

$$\boldsymbol{t} = \mathrm{d} \cdot \boldsymbol{u} / \| \boldsymbol{u} \|. \tag{17}$$

The geometric model of the object provides the parameters of the rigid transform between the vertex-centered frame and the object-centered frame, i.e., A_2.

In conclusion, a junction-to-vertex assignment constrains five of the six degrees of freedom associated with the position and orientation of an object. If two such assignments are available, the sixth parameter d may be easily computed. Let J_1/V_1 and J_2/V_2 be two junction-to-vertex assignments (see Fig. 7). Let $\boldsymbol{e}$ be the vector from V_1 to V_2, $\boldsymbol{u}_1$ be the vector from F (the focal point) to J_1, and $\boldsymbol{u}_2$ be the vector from F to J_2. The distance from F to V_1, say d_1 is given by the expression:

$$\mathrm{d}_1 = \| \boldsymbol{u}_1 \| \, \| \boldsymbol{u}_2 \times \boldsymbol{e} \| / \| \boldsymbol{u}_1 \times \boldsymbol{u}_2 \|. \tag{18}$$

D. The Matching Strategy

As we have mentioned earlier, the matching process consists of finding a unique correspondence between a set of image features and a set of object features. To do that we use a hypothesize-and-test strategy implemented as a depth-first tree search. This algorithm is paraphrased below.

Hypothesis generation: Select a few image features such that when they are put into correspondence with model features they constrain the position and orientation of the part; compute the object-to-camera transform implied by this assignment and predict where to look for other features.

Hypothesis verification: For each predicted object feature, the list of image features is checked to see whether any has the predicted qualities; if more than one image feature has the predicted qualities select one of them; use its measured position and orientation to refine the object-to-camera transform associated with the current hypothesis; extend the current match to include this feature.

The verification step is repeated for all the predicted object features until either a satisfactory match is found or until the algorithm fails to observe a predicted feature. In the latter case, the algorithm backtracks to the last choice point. Choice points arise when more than one image feature appears in a predicted location. If at some stage of the search there is no choice point and if a satisfactory match has not yet been found, the algorithm skips over a predicted feature. This allows the procedure to deal with objects that are partially visible and/or with imperfect data. If there is not enough evidence for verifying a hypothesis, a new hypothesis is generated and the same verification procedure is repeated.

This is the outline of the algorithm. It is a *grow-a-match* approach which has already been used for a wide range of applications: two-dimensional object recognition from binary images [15] (the LFF system), three-dimensional object recognition from dense range data [16] (the 3DPO system), from sparse range data or tactile data [20], or from intensity data [12]. Let us describe the specific implementation when the hypothesize-and-test strategy is associated with the backprojection of image linear features.

The image figure which is used for generating a hypothesis is a three-line image junction while any of the image lines, angles or junctions may be used for verification. An image junction which has not yet been included in a match is selected from the image. If this figure is being interpreted as an object vertex, there are six possible matches between the lines of the junction l_1, l_2, l_3 and the edges of the vertex, $\boldsymbol{L}_1$, $\boldsymbol{L}_2$, $\boldsymbol{L}_3$:

- Three matches if the edges are scanned clockwise:

$$(l_1/L_1, l_2/L_2, l_3/L_3), (l_1/L_2, l_2/L_3, l_3/L_1),$$
$$(l_1/L_3, l_2/L_1, l_3/L_2),$$

- Three matches if the edges are scanned counterclockwise:

$$(l_1/L_1, l_2/L_3, l_3/L_2), (l_1/L_3, l_2/L_2, l_3/L_1),$$
$$(l_1/L_2, l_2/L_1, l_3/L_3),$$

The two sets of solutions correspond to the two spatial solutions of a backprojected junction (Section II-D). The algorithm considers one such pairing and computes 5 of the 6 degrees of freedom of the object in camera coordinates. Let d_{min} and d_{max} be the lower and upper bounds of

the missing parameter, i.e., the distance from the focal point to the object vertex. The ensemble of these parameters constitute the current hypothesis: exact values for the three rotations and bounds for the three translations. Hence, there are two matrices associated with a hypothesis, A_{min} and A_{max} which have the same rotational parameters. Based on these parameters a set of visible vertices and a set of visible edges can be computed. Moreover, the *image appearance* of each visible vertex can be predicted as well: if only one adjacent face of the vertex is visible, the appearance will be an angle and if two or three faces are visible, the appearance will be a junction. This works exactly like a hidden surface elimination algorithm.

The hypothesis verification stage attempts to find how many of the predicted object features match image features and to refine the 6 degrees of freedom. Let us consider a vertex from the list of visible vertices associated with the *current hypothesis*. The tip of this vertex projects onto the image within a rectangular region. The size of this region depends on the values of d_{min} and d_{max}. Let V be the tip of this vertex; its model (homogeneous) coordinates are $(V_x, V_y, V_z, 1)$. Then the diagonal of the rectangular image region is defined by the projections of V, say V_{p1} and V_{p2}. V_{p1} is obtained using A_{min} and V_{p2} is obtained using A_{max}. For example, the coordinates of V_{p1} are $(x_1 f/z_1, y_1 f/z_1, f, 1)$ where x_1, y_1, and z_1 are given by:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ s \end{pmatrix} = A_{min} \begin{pmatrix} V_x \\ V_y \\ V_z \\ 1 \end{pmatrix}. \tag{19}$$

Next the algorithm collects image figures whose tips belong to this region and which have not yet been included in a match. If the predicted appearance of the vertex is a junction, the algorithm collects junctions, angles, and lines simply because the segmentation process is not perfect and some features may have not been detected. Next, one of these image features is selected and its position is used to compute an exact value for the depth [equation (18)]. A unique matrix A associated with the current match can be derived. The algorithm checks now if the lines of the image figure being considered do correspond to the edges of the predicted vertex. Each predicted edge together with the focal point defines a plane, say a *verification plane*. An image line that matches this edge must belong to its verification plane. Therefore the image-figure-to-object-vertex assignment is either accepted or rejected. In the first case (accepted), the actual position and orientation of the image figure are used to refine the rotational parameters and to evaluate new tighter bounds for the translational parameters and the current match is expanded to include this assignment. In the latter case (rejected), the algorithm backtracks to the last choice point.

Choice points appear whenever a predicted feature has more than one potential match. If, at some stage of the search there is no choice point, there are two alternatives: either to skip over the vertex or to reject the hypothesis and generate a new one.

The expensive computation associated with the back-projection of a junction is performed only by the hypothesis generation step of the algorithm. Moreover, as it has been explained in Section II-D, most of this computation is carried out only once for each junction. In fact, the verification step does not even need at all to deal with junctions; it is sufficient to detect one junction per object. It is therefore important to select the junctions carefully.

One open issue with this type of algorithms is the evaluation of the minimum number of predicted features that need to be verified in order to accept a good match, i.e., an object detection. If such a minimum set were available, the search could be abandoned any time a sufficient number of mutually consistent assignments were found. Otherwise exhaustive search has to be performed for every hypothesis.

It is hard to evaluate the cost of the algorithm for two main reasons. First, because the image data contain a lot of irrelevant features such as shadows or texture noise and second, because the computation associated with each node is not homogeneously distributed across the search space. Hypothesis generation is more expensive than verification. Furthermore, as the algorithm expands a match, the locational parameters are more and more accurate and hence the number of image feature candidates for verifying a predicted object feature decreases.

E. Using More Knowledge

There are two kinds of knowledge which may be used for speeding up the search process: prior knowledge about the configuration of the scene being viewed or knowledge derived directly from the image.

In the first case, one of the basic assumptions is that the orientation of the *support plane* or ground plane relative to the camera is known. If the objects to be located and recognized lie in a stable position onto this plane, three out of the six degrees of freedom are constrained to a set of discrete values whose size depends on the number of stable positions of each object. This can dramatically reduce the search space since hypothesis generation needs to specify only three parameters (two translations and one rotation). Nevertheless, this type of reasoning cannot be applied to objects jumbled together in a bin.

As with all tree searches it is important to order in advance the alternatives according to their expected utility. To minimize execution time, it is possible to order the alternatives in advance. However, here we concentrate on the use of knowledge derived at runtime from the image.

So far we have described the search process on the basis of isolated groups of image features such as angles and junctions. If two such groups share a common feature, they are inherently less ambiguous than isolated groups. Consider for example two junctions sharing a common line. Assume that a vertex has been assigned to the *first*

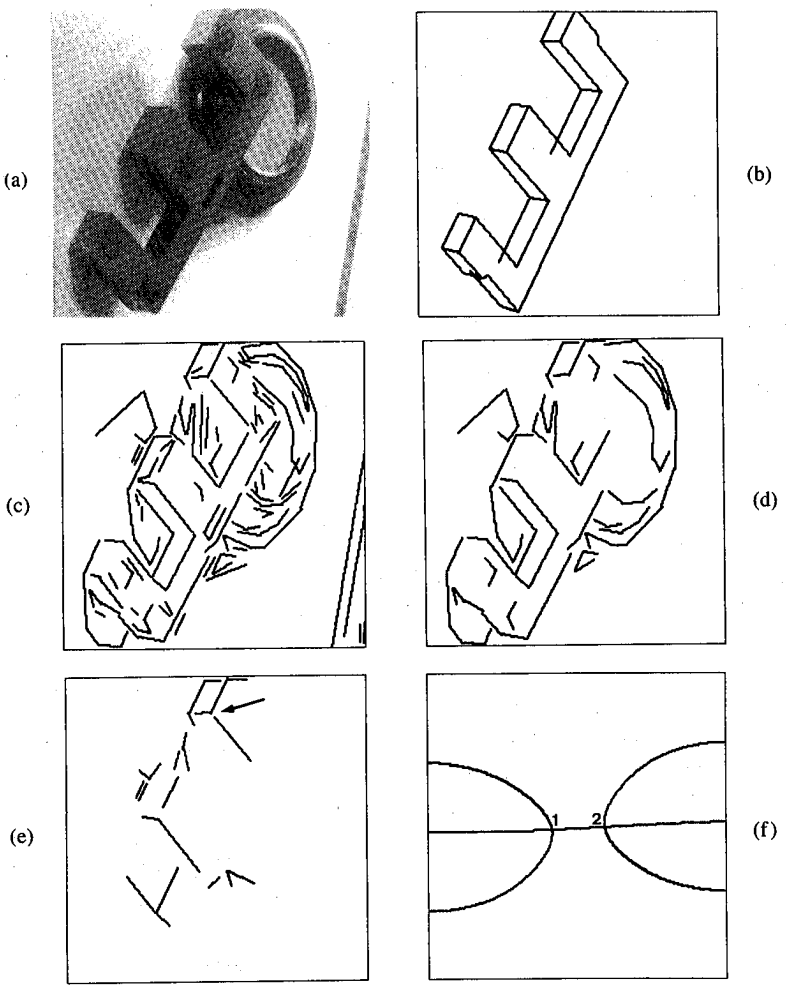

Fig. 8. (a) The gray-level image of a part taken with a 90 mm lens. (b) The final result of recognition. (c) Lines. (d) Angles. (e) Junctions. (f) Backprojection of the junction shown by an arrow being interpreted as a right vertex.

junction in this pair and hence a hypothesis is generated. Moreover, a set of visible vertices is predicted. Hypothesis verification should consider first those visible vertices which share common edges with the initial vertex (the one assigned to the first junction) and should attempt to match one of these visible vertices to the *second junction* in the pair.

Similarly, polygon finding may help out the selection of the search space node the best suited for expansion. Whenever an image junction shares two common lines with a polygon, there is strong positive evidence that one of the faces adjacent to the vertex being assigned to this junction is visible and hence the matching algorithm should try to assign one of the object's faces to this image polygon.

V. Experimental Results

Let us show how this recognition algorithm works in practice. Fig. 8(a) shows the image of an industrial part leaning on another object. This image has been taken with a television camera using a 90 mm lens. The part is at approximately 0.9 m away from the focal point. With this setting there is almost no perspective distortion and hence, techniques which rely on detecting vanishing points cannot be used. The position and the orientation of the table top relatively to the camera is not known. Fig. 8(b) shows the final result of recognition. The wireframe model of the part is displayed using the object-to-camera transform computed by the matching process:

$$A = \begin{pmatrix} -0.78 & -0.02 & 0.63 & -5.57 \\ -0.61 & 0.27 & -0.74 & -1.48 \\ -0.16 & -0.96 & -0.22 & 773.6 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (20)$$

Fig. 8(c) shows the result of line detection. Fig. 8(d) and (e) shows the result of angle and three-line junction detection. The junction shown by an arrow has been considered first by the matching algorithm because it shares two lines with a four-line image polygon and because it has the longest lines. If this junction is being interpreted as a right vertex, there are two solutions for its spatial orientation.

The algorithm selects a convex interpretation first because convex edges are more likely to appear in the data. The solution 2 in Fig. 8(f) corresponds to a convex vertex. There are 16 such vertices in the model and there are 3 ways to combine the lines of the junction with the edges of a vertex. Hence, there are 48 possible matches. However, the presence of an image polygon adjacent to the junction allows to rapidly eliminate most of these matches. The remaining ones (12 matches) are those which have correctly predicted the image size and orientation of the polygon. These 12 hypotheses are further expanded. At the end of an exhaustive search a set of 13 mutually consistent assignments (including the hypothesis) is selected as the best available match.

The computation time of an implementation of the hypothesis-and-test algorithm is very small (1/10) compared to the time required for preprocessing: line, angle, junction, and polygon finding and precomputation of junction backprojection, i.e., the evaluation of the terms f, g_1, g_2, and g_3 in (11) and (12).

VI. Discussion

We have devised and implemented a new method for retrieving three-space properties from a single perspective view. We have been inspired from a psychophysical model for spatial reasoning. The method backprojects image features onto the 3-D space and combines geometric constraints with heuristic assumptions to find feasible solutions. A model-based object recognition scheme has been implemented as a search strategy. The difference between this technique and previous approaches is that it performs the model-to-image matching in three-space rather than predicting object appearances in the image.

The mathematics of backprojection are relatively simple in the case of straight-lines. Unfortunately, this is not true for higher-order curves. Barnard [4] and Brady and Yuille [6] offer some suggestions. An alternative approach could be to derive 3-D constraints from an extended catalog of feature groupings such as second-order curves and lines.

Image feature grouping is necessary for reducing the search space: a group of features has fewer interpretations than isolated ones and therefore the combinatorial explosion of the search space is reduced. However, it is not easy to perform this grouping and to partially solve for the ambiguity of the interpretation tree in the image plane. Only ad hoc techniques exist so far and one goal of our research is to devise a general purpose image feature grouping method. Feature grouping may provide the first index into a memory of shapes for model retrieval.

We also plan to extend our approach to be able to deal with a sequence of images. If the geometric rigid structure undergoing a scene could be related to every image in the sequence, then the complexity of the image-to-image correspondence problem may be reduced. This type of constraint is equally valid for a stereo pair and for time-varying imagery. Moreover, the viewer-to-scene relation derived from monocular analysis may be useful for calibrating a camera position with respect to another position if the camera were mounted on a mobile robot.

Appendix
Derivation of Equation (10)

Let us replace in (9) the expression of $\boldsymbol{L}_3$ given by (8). The unit vector $\boldsymbol{L}_1$ is given by the cross-product of $\boldsymbol{S}$ and $\boldsymbol{P}_1$ divided by the norm of the cross-product, i.e., (4). We obtain:

$$\boldsymbol{S} \cdot \boldsymbol{P}_3 \| \boldsymbol{S} \times \boldsymbol{P}_1 \| \sin \psi + (\boldsymbol{S} \times \boldsymbol{P}_1) \cdot \boldsymbol{P}_3 \cos \theta \cos \psi + (\boldsymbol{S} \times (\boldsymbol{S} \times \boldsymbol{P}_1)) \cdot \boldsymbol{P}_3 \sin \theta \cos \psi = 0 \quad (21)$$

Using the following known equalities:

$$\boldsymbol{S} \times (\boldsymbol{S} \times \boldsymbol{P}_1) = (\boldsymbol{S} \cdot \boldsymbol{P}_1)\, \boldsymbol{S} - \| \boldsymbol{S} \| \boldsymbol{P}_1 \quad (22)$$

$$(\boldsymbol{S} \times \boldsymbol{P}_1) \cdot \boldsymbol{P}_3 = \boldsymbol{S} \cdot (\boldsymbol{P}_1 \times \boldsymbol{P}_3) \quad (23)$$

and noticing that $\boldsymbol{S}$ is a unit vector, one may easily derive (10).

Acknowledgment

The author would like to thank M. Dhome from the Université de Clermont-Ferrand and P.-L. Borianne from LIFIA for their useful comments on a first draft of this paper.

References

[1] D. Marr, *Vision.* San Francisco, CA: W. H. Freeman, 1982.
[2] L. A. Cooper and R. N. Shepard, "Turning something over in the mind," *Sci. Amer.*, pp. 114–120, Dec. 1984.
[3] T. Kanade, "Recovery of the 3d shape of an object from a single view," *Artificial Intell.*, vol. 17, pp. 409–460, Aug. 1981.
[4] S. T. Barnard, "Interpreting perspective images," *Artificial Intell.*, vol. 21, pp. 435–462, Nov. 1983.
[5] ——, "Choosing a basis for perceptual space," in *Workshop Comput. Vision, Representation and Control*, Annapolis, MD, Apr. 1984, pp. 225–230.
[6] M. Brady and A. Yuille, "An extremum principle for shape from contour," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 288–301, May 1984.
[7] A. P. Witkin, "Recovering surface shape and orientation from texture," *Artificial Intell.*, vol. 17, pp. 17–45, Aug. 1981.
[8] J. R. Kender, "Environmental relations in image understanding: The force of gravity," in *Proc. Image Understanding Workshop*, Arlington, VA, June 1983, pp. 249–256.
[9] R. Horaud, "Combining image and spatial reasoning for model retrieval," in *Proc. 7th European Conf. Artificial Intell.*, Brighton, England, July 1986, pp. 529–538.
[10] R. L. Gregory, *The Intelligent Eye.* New York: McGraw-Hill, 1974.
[11] I. Rock, *The Logic of Perception.* Cambridge, MA: M.I.T. Press, 1983.
[12] C. Goad, "Special purpose automatic programming for 3d model-based vision," in *Proc. Image Understanding Workshop*, Arlington, VA, June 1983, pp. 94–104.
[13] R. A. Brooks, "Symbolic reasoning among 3d models and 2d images," *Artificial Intell.*, vol. 17, pp. 285–348, Aug. 1981.
[14] D. Lowe, "Visual recognition from spatial correspondence and perceptual organization," in *Proc. 9th Int. Joint Conf. Artificial Intell.*, Los Angeles, CA, Aug. 1985, pp. 953–959.
[15] R. C. Bolles and R. A. Cain, "Recognizing and locating partially visible objects, the local-feature-focus method," *Int. J. Robotics Res.*, vol. 1, no. 3, pp. 57–82, 1982.

[16] R. Horaud and R. C. Bolles, "3dpo's strategy for matching 3d objects in range data," in *Proc. IEEE Int. Conf. Robotics*, Atlanta, GA, Mar. 1984, pp. 78–85.
[17] D. Marr and E. Hildreth, "Theory of edge detection," M.I.T. Artificial Intelligence Lab., A. I. Memo. 518, Apr. 1979.
[18] T. Pavlidis and S. L. Horowitz, "Segmentation of plane curves," *IEEE Trans. Comput.*, vol. C-23, pp. 860–870, Aug. 1974.
[19] R. C. Bolles, R. Horaud, and M. J. Hannah, "3dpo: A three-dimensional part orientation system," in *Proc. 8th Int. Joint Conf. Artificial Intell.*, Karlsruhe, Germany, Aug. 1983, pp. 1116–1120.
[20] W. E. L. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *Int. J. Robotics Res.*, vol. 3, no. 3, pp. 3–35, 1984.

Radu Horaud was born in Bucharest, Rumania, on August 13, 1953. He emigrated to France in 1972. He received both the Diplome d'Ingénieur and the Diplome de Docteur-Ingénieur degrees from the Ecole Nationale Supérieure d'Ingénieurs Electriciens de Grenoble, France, in 1977 and 1981, respectively.

He spent two years (1982–1984) as an International Fellow at SRI International in the Department of Robotics. He is currently a Research Scientist at the Laboratoire d'Informatique Fondamentale et d'Inelligence Artificielle (LIFIA) Grenbole, France. At the moment his research interests include computer vision, computational geometry, 3-D object recognition, and modeling.