3-D Shape Recovery Using Distributed Aspect Matching

Sven J. Dickinson, Alex P. Pentland, and Azriel Rosenfeld, *Fellow, IEEE*

Abstract— **We present an approach to the recovery of 3-D volumetric primitives from a single 2-D image. The approach first takes a set of 3-D volumetric modeling primitives and generates a hierarchical aspect representation based on the projected surfaces of the primitives; conditional probabilities capture the ambiguity of mappings between levels of the hierarchy [15]. From a region segmentation of the input image, we present a novel formulation of the recovery problem based on the grouping of the regions into aspects. No domain-dependent heuristics are used; we exploit only the probabilities inherent in the aspect hierarchy. Once the aspects are recovered, we use the aspect hierarchy to infer a set of volumetric primitives and their connectivity.**

As a front end to an object recognition system, the approach provides the indexing power of complex 3-D object-centered primitives while exploiting the convenience of 2-D viewer-centered aspect matching. However, unlike traditional aspect matching paradigms that represent the entire object with a set of aspects, we use aspects to represent a finite vocabulary of 3-D parts from which objects can be constructed. Thus, the size of our aspect set is fixed and, more important, independent of the size of the object database. The method not only fully accommodates occlusion but uses the aspect hierarchy to overcome image segmentation errors. We describe the approach in detail and demonstrate its application to both synthetic line drawings and real images.

Index Terms— **Aspect modeling, geons, recognition by parts, 3-D object recognition, 3-D shape recovery, volumetric object modeling primitives.**

I. Introduction

SIGNIFICANT progress has been made in the feature-based recognition of 3-D objects from 2-D images; some important examples include Lowe [33], Huttenlocher and Ullman [27], Thompson and Mundy [59], and Lamdan *et al.* [31]. However, these approaches restrict the features to simple 2-D primitives such as line segments, corners, inflections, and 2-D perceptual structures. These primitives are appealing because of their viewpoint invariance. However, due to their simplicity, a typical 3-D model contains a large number of primitives. Consequently, the process of searching a large database to recognize a model becomes inefficient. Furthermore, the simplicity of the primitives makes recognition unreliable, and detailed verification of the model's pose is required. Such verification is not only expensive but restricts the recognition system to models whose exact geometry is known beforehand.

Our approach is to use more complex primitives so that indexing for recognition is efficient, and only qualitative (topological) verification is required. We have chosen to model objects as configurations of object-centered 3-D volumetric primitives such as polyhedra, generalized cylinders, and superquadrics. This approach shifts the burden of recognition from the top-down verification of simple 2-D features to the bottom-up extraction and grouping of features into volumetric primitives. The recovery of these 3-D primitives would normally entail a high search cost due to the their complexity. However, we have been able to avoid this problem by taking advantage of probabilistic information about whatever set of modeling primitives the user has chosen.

To obtain the probabilistic information, we choose a set of primitives and map them into a set of viewer-centered aspects whose size is fixed and *independent* of the size of the object database [15]. The aspects are represented by a hierarchy of 2-D features whose levels include the qualitative shapes of the primitives' projected surfaces (faces), subsets of the contours that bound the faces (boundary groups), and groups of faces (aspects). The relations between these features are then assessed from *all* viewpoints, thus generating a table of estimated conditional probabilities for each feature and primitive as a function of less complex features. For instance, one entry in this table might be the conditional probability that we are viewing a cylinder primitive given that we have found a rectangular face in the image.

Given an image of a scene, this table of conditional probabilities is then used to guide a combinatorial search that yields a full and consistent interpretation of the viewed scene. The key idea is that the statistical properties of the set of user-defined primitives are used to avoid a combinatorial explosion in the search process. Knowledge about how each primitive looks from all angles makes for a more informed search and allows the use of much more complex indexing features than are typically employed. The use of such complex features and primitives establishes the foundation for a more robust recognition system: one that can potentially accommodate unexpected objects [48].

The organization of the paper is as follows. In Section II, we present our hybrid representation for object modeling which integrates object-centered and viewer-centered models. Section III presents our volumetric primitive recovery algorithm,

Manuscript received October 30, 1990; revised May 6, 1991. This work was supported by the Air Force Office of Scientific Research under Grant AFOSR-86-0092.

S. J. Dickinson was with the Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD 20742-3411. He is now with the Department of Computer Science, University of Toronto, Toronto, Canada M5S 1A4.

A. P. Pentland is with the Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139.

A. Rosenfeld is with the Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD 20742-3411.

IEEE Log Number 9102683.

whereas Section IV explores the sensitivity of the algorithm to segmentation errors. We have written a computer program to demonstrate the approach, and in Section V, we apply the program to both synthetic line drawings and real images. In Section VI, we contrast our approach to related work and conclude with a discussion in Section VII.

II. Building the Search Tables

A. Choosing the 3-D Primitives

Given a database of object models representing the domain of a recognition task, we seek a set of 3-D volumetric primitives that, when assembled together, can be used to construct the object models. Many 3-D object recognition systems have successfully employed 3-D volumetric primitives to construct objects. Commonly used classes of volumetric primitives include polyhedra [47], generalized cylinders [9], and superquadrics [39]. Whichever set of volumetric modeling primitives is chosen, they will be mapped to a set of viewer-centered aspects.

Consider, for example, a rectangular block primitive that might be a component of many objects in a database. Let us assume that for each object of which it is a component, its dimensions are different. If our aspect definitions were *quantitative*, specifying the exact geometry of image features, each instance of the block would map to a different set of aspects. However, if the aspect definitions were *qualitative*, providing stability under minor changes in the shape of the primitives (e.g., scale, dimension, and curvature), a single set of aspects might represent all possible instances of a rectangular block. Our approach, therefore, has been to select a set of qualitatively defined volumetric primitives so that their description will be invariant under such changes in shape.

1) The Primitives: To demonstrate our approach to primitive recovery, we have selected an object representation similar to that used in Biederman's recognition by components (RBC) theory [8]. RBC suggests that from nonaccidental relations in the image, a set of contrastive dichotomous (e.g., straight versus curved axis) and trichotomous (e.g., constant versus tapering versus expanding/contracting cross-sectional sweep) 3-D primitive properties can be determined. The Cartesian product of the values of these properties give rise to a set of volumetric primitives called geons. Biederman's geons constitute only one possible selection of qualitatively defined volumetric primitives; the general approach of applying the Cartesian product to a set of contrastive primitive properties can be used to generate many different volumetric primitive representations. For our investigation, we have chosen three properties including cross-section shape, axis shape, and cross-section sweep. The values of these properties give rise to the following set of ten qualitative volumetric primitives[1]:

1) Rectangular cross-section, straight axis, and constant cross-section size

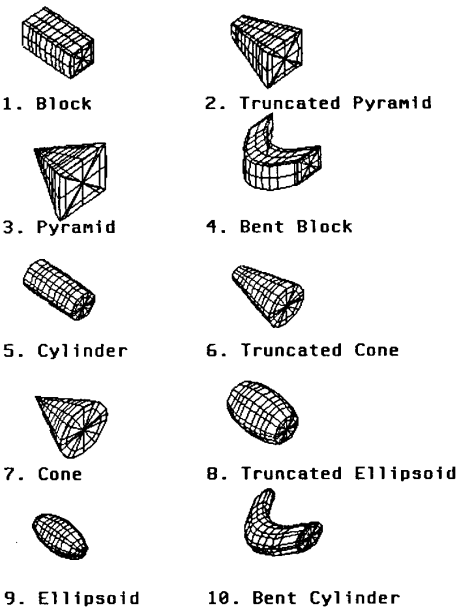

Fig. 1. Ten primitives.

2) rectangular cross-section, straight axis, and linearly increasing cross-section size not starting from a point
3) rectangular cross-section, straight axis, and linearly increasing cross-section size starting from a point
4) rectangular cross-section, curved axis, and constant cross-section size
5) elliptical cross-section, straight axis, and constant cross-section size
6) elliptical cross-section, straight axis, and linearly increasing cross-section size not starting from a point
7) elliptical cross-section, straight axis, and linearly increasing cross-section size starting from a point
8) elliptical cross-section, straight axis, and ellipsoidally increasing then decreasing cross-section size neither starting from nor ending at a point
9) elliptical cross-section, straight axis, and ellipsoidally increasing then decreasing cross-section size starting from and ending at a point
10) elliptical cross-section, curved axis, and constant cross-section size.

In our system, these ten primitives were modeled using Pentland's SuperSketch 3-D modeling tool [39], as illustrated in Fig. 1.[2] We believe that this taxonomy of volumetric primitives is sufficient to model a large number of objects; however, nothing in our approach is specialized for this particular set of primitives. If necessary, our approach can easily accommodate other sets of volumetric primitives.

2) Primitive Attachment: Having defined a set of modeling primitives, we must decide how to connect them to construct objects. We have adopted a qualitative convention based on a labeling of each primitive's attachment surfaces. For

[1] The Cartesian product of the values of these properties results in a set of 20 primitives; however, to simplify the investigation in terms of generating the conditional probability tables described in the next section, we have chosen a subset of 10 primitives.

[2] SuperSketch models each primitive with a superquadric surface that is subjected to bending, tapering, and pinching deformations.

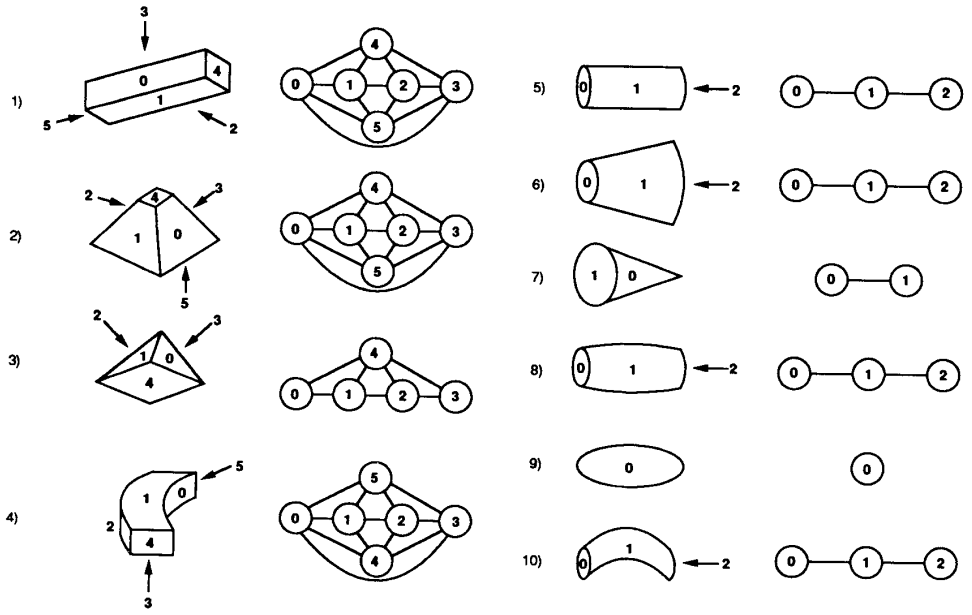

Fig. 2. Primitive attachment surface assignments.

example, the truncated cone primitive (primitive 6) has three attachment surfaces: the small end, the large end, and the side. Similarly, the curved block primitive (primitive 4) has six attachment surfaces: the concave side, the convex side, the two planar sides, and the two planar ends. The attachment surface labels for the ten primitives can be found Fig. 2. In the current implementation, we have restricted any junction of two primitives to involve exactly one attachment surface from each primitive.

3) Extension to Include Quantitative Information: Although our current system uses only qualitative geometric information, it is straightforward to include quantized metric information. For example, properties such as cross-section extent and axis curvature can provide important cues for recognition even though they are not viewpoint invariant and thus not always available. Such metric information can be included in our symbolic matching process by partitioning image measurements into coarse bins (e.g., an axis might be "slightly curved" or "strongly curved"). Similarly, we can also coarsely specify the position of a join between two surfaces (e.g., is the attachment near the middle or the corners of a face?) and the angles at which they join (e.g., acute or approximately perpendicular).

B. Defining the 2-D Aspects

Traditional aspect graph representations of 3-D objects model an entire object with a set of aspects, each defining a topologically distinct view of an object in terms of its visible surfaces [30]. Our approach differs in that we use aspects to represent a (typically small) set of volumetric primitives from which each object in our database is constructed, rather than representing an entire object directly. Consequently, our goal is to use aspects to recover the 3-D primitives that make up the object in order to carry out a recognition-by-parts procedure, rather than attempting to use aspects to recognize entire objects. The advantage of this approach is that since the number of qualitatively different primitives is generally small, the number of possible aspects is limited and, more important, *independent* of the number of objects in the database. The disadvantage is that if a primitive is occluded from a given 3-D viewpoint, its projected aspect in the image will also be occluded. Thus, we must accommodate the matching of occluded aspects, which we accomplish by use of a hierarchical representation we call the *aspect hierarchy*.

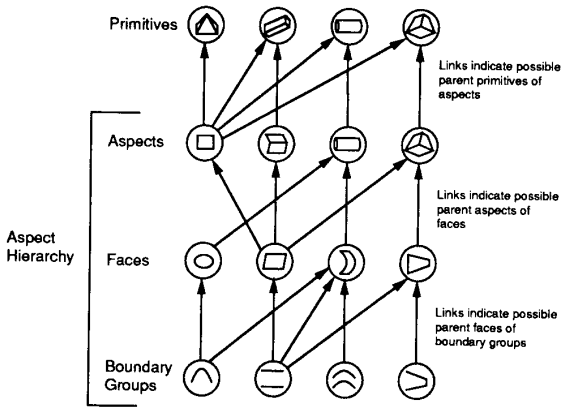

Fig. 3. Aspect hierarchy.

The aspect hierarchy consists of three levels based on the faces appearing in the aspect set:

- *Aspects* constitute the top level of the aspect hierarchy and represent the set of topologically distinct views of the primitives; each aspect consists of a set of 2-D faces, each corresponding to a primitive surface. Identification of the aspects can allow identification of the visible primitives.

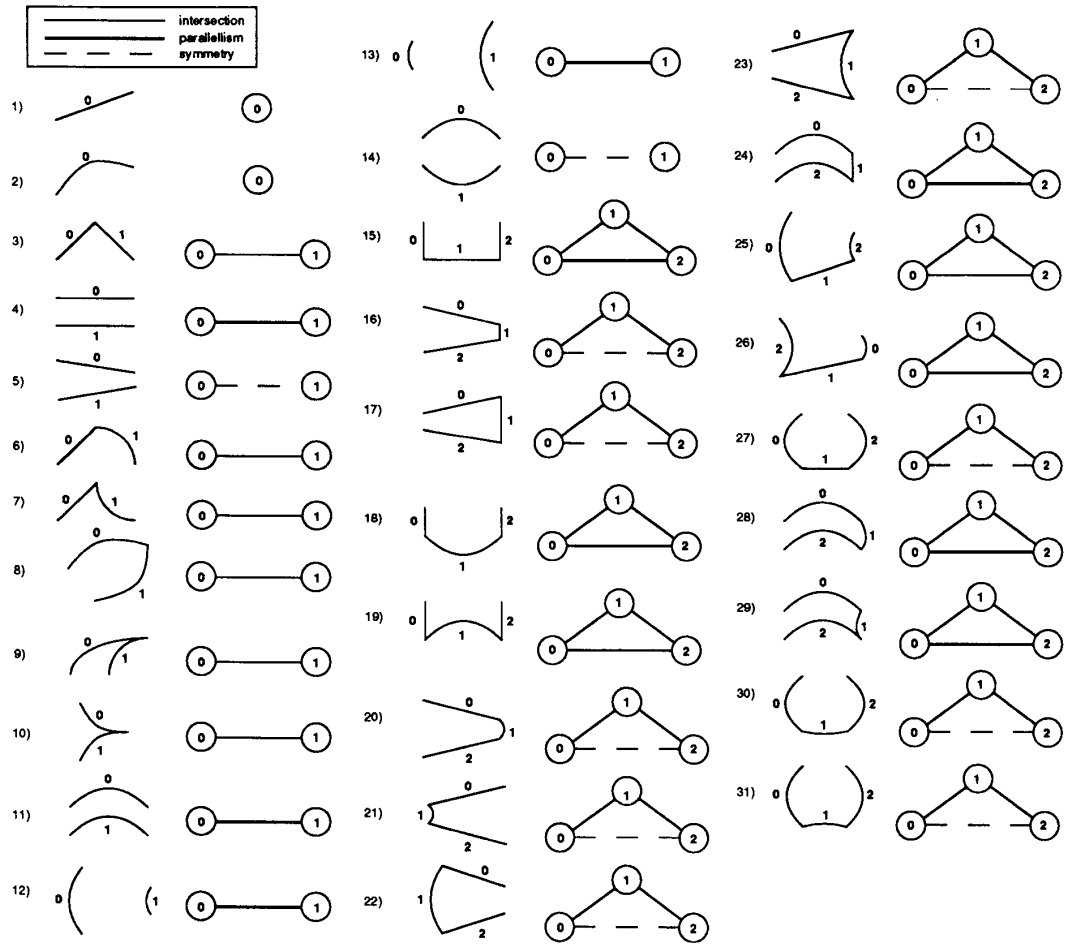

Fig. 4. Boundary groups.

However, due to occlusion, some of the faces in an aspect may be partially or completely missing. When this occurs, we may need to analyze the arrangement of the remaining faces, and therefore, we introduce the second level of the aspect hierarchy.

- *Faces* that make up the various aspects form the second level of the aspect hierarchy. Reasoning about the type and arrangement of visible faces can allow identification of an aspect even when it is partially occluded. However, again due to occlusion, some of the contours that make up a face may be partially or completely missing. When this occurs, we may need to analyze the arrangement of the remaining contours, and therefore, we introduce the lowest level of the aspect hierarchy.
- *Boundary groups* are subsets of the faces' bounding contours and make up the third and lowest level of the aspect hierarchy. The boundary groups provide a mechanism for identifying the face type even when the face is partially occluded.

Fig. 3 illustrates a portion of the aspect hierarchy, while the following describe the levels of the aspect hierarchy in more detail.

Boundary Groups: Boundary groups represent all subsets of lines and curves comprising the faces; the complete set of 31 boundary groups can be found in Fig. 4. A boundary group is represented by a graph in which nodes represent bounding contours and arcs represent certain nonaccidental contour relations, including parallelism, symmetry, and intersection, that occur within a particular face. The use of these nonaccidental relationships for extracting larger-scale structure has been suggested by many authors, including Witkin and Tenenbaum [61], Lowe [33], and Biederman [8]. These boundary groups represent *qualitative* relationships among *qualitatively described* contours; exact lengths, distances, angles, curvature, etc., are not represented.

Faces: Faces represent the set of component faces appearing in the aspects; the complete set of 16 faces can be found in Fig. 5. A face is represented by a graph in which nodes represent contours bounding the face and arcs represent relations between the contours. Each face differs in the number of constituent contours, the types of contours (e.g., straight, concave, or convex), or the nonaccidental relations between the contours.

Aspects: Aspects are connected sets of faces; Fig. 6 con-

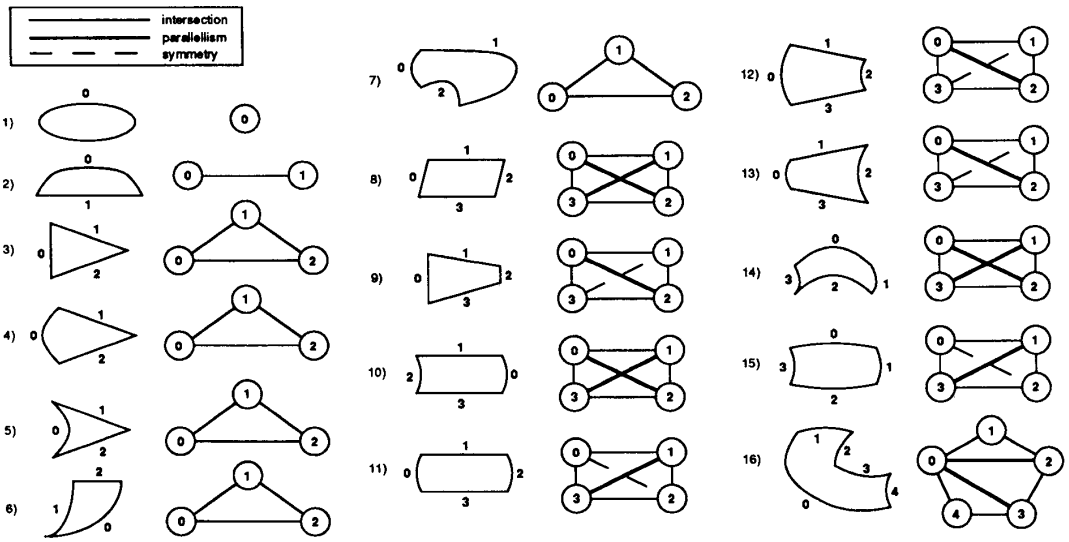

Fig. 5. Faces.

tains the complete set of 37 aspects. An aspect is represented by a graph in which nodes represent faces and arcs represent face adjacencies; arc labels indicate those contours shared by adjacent faces.

C. Relating the 2-D Aspects to the 3-D Primitives

A given boundary group may be common to a number of faces. Similarly, a given face may be a component of a number of aspects, while a given aspect may be the projection of a number of primitives. To capture these ambiguities, we have created a matrix representation that describes conditional probabilities associated with the mappings from boundary groups to faces, faces to aspects, and aspects to primitives. For example, consider the mapping between faces and aspects. To describe this mapping, we create a matrix whose rows represent faces and whose columns represent aspects. If a particular face can be a component of ten different aspects, then those ten column entries corresponding to the ten aspects contain a value from 0 to 1.0, indicating the probability that the face is part of that particular aspect. Thus, the entries along each row sum to 1.0. Fig. 7 presents a portion of the aspect hierarchy and related primitives along with the corresponding portions of the matrices.

To generate these conditional probabilities for the boundary group to face, face to aspect, and aspect to primitive mappings, we use the following procedure. We first model our 3-D volumetric primitives using the SuperSketch modeling tool [39]; the modeled primitives are shown Fig. 1. The next step in generating the probability tables involves rotating each primitive about its internal x, y, and z axes in 10° intervals. The resulting quantization of the viewing sphere gives rise to 648 views per primitive; however, by exploiting primitive symmetries, we can reduce the number of views for the entire set of primitives to 688. For each view, we orthographically project the primitive onto the image plane and note the appearance of each feature (boundary group, face, and aspect) and its parent. The resulting frequency distribution gives rise to the three conditional probability matrices (which can be found in [16]). This procedure implicitly assumes that all primitives are equally likely to appear in the image and that all spatial orientations are equally likely. However, if a set of *a priori* probabilities of occurrence or orientation are known, they can be incorporated into the analysis by simply including them when forming the frequency distribution tables. In our current system, the process of identifying and counting the features in each projection is not yet fully automated; however a number of algorithms exist for automatically computing aspect graphs under certain restrictions [24], [25], [19], [52], [54], [55].

D. Discussion

An alternative approach to our hierarchical aspect representation would be to map the features at the lowest level (in our case, the boundary groups) directly to the 3-D models (in our case, the 3-D primitives). This approach is advocated, for example, by Lowe [33], Huttenlocher and Ullman [27], and Lamdan *et al.* [31]. In such a scenario, a given boundary group index would return a set of candidate primitives containing that boundary group. The drawback of this approach is that simple contour-level descriptions seem to provide only weak constraints for identifying primitive type and orientation.

Considerable insight can be gained by a careful examination of the three conditional probability matrices described above. By multiplying together matrices representing adjacent levels of the aspect hierarchy, we can generate new matrices that describe the mappings between boundary groups and aspects, boundary groups and primitives, and faces and primitives. These three new mappings, along with the three original mappings, can best be displayed as a set of histograms. To generate these histograms, we first determine the largest conditional probability between each low-level feature and a

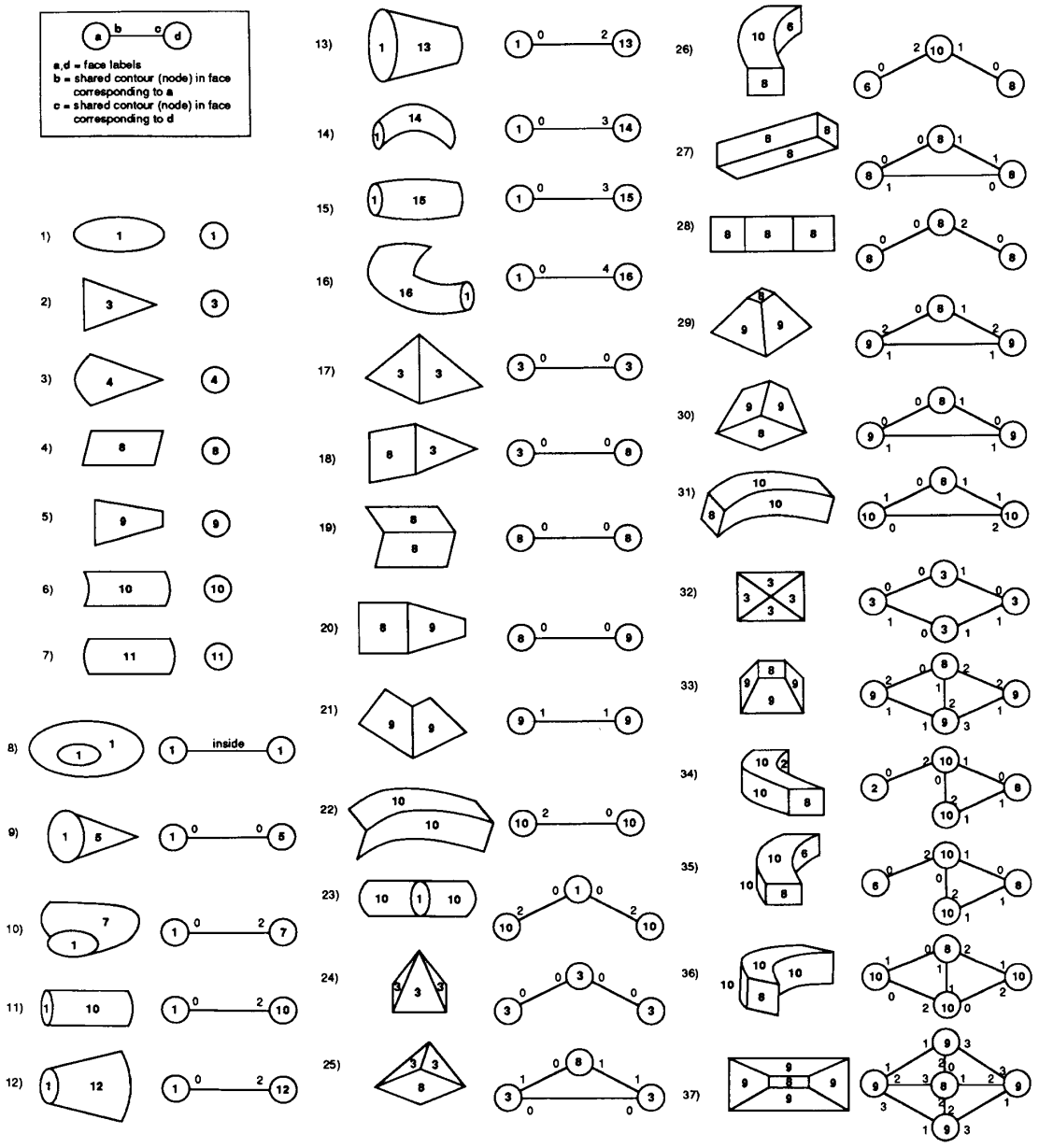

Fig. 6. Aspects.

higher-level feature, thus obtaining a measure of the ambiguity in that mapping. For example, a node having two emanating arcs with values 0.50 and 0.50 is inferentially inferior to a node having three emanating arcs with values 0.90, 0.05, and 0.05. Having found the highest probability mapping associated with each low-level feature, these probabilities are histogrammed into ten equal bins. The resulting histograms, presented in Fig. 8, show the percentage of features whose highest probability mapping falls within a given probability range.

In this manner, the mappings from aspects, faces, and boundary groups to the primitives can be examined; these three histograms are shown in Figs. 8(c), (e), and (f), respectively. In addition to providing maximum contraint on primitive orientation, the aspect to primitive mapping is the strongest, with 90% of the aspects having a high conditional probability (0.80–1.0) mapping. The mappings from the faces and boundary groups to the aspects can also be examined; these two histograms are shown in Figs. 8(b) and (d), respectively. In this case, the mapping from faces to aspects is much less ambiguous than the mapping from boundary groups to aspects. The remaining possible mapping, from boundary groups to faces, is shown in Fig. 8(a).

The aspect hierarchy effectively *prunes* the mapping from boundary groups to primitives by introducing topological and

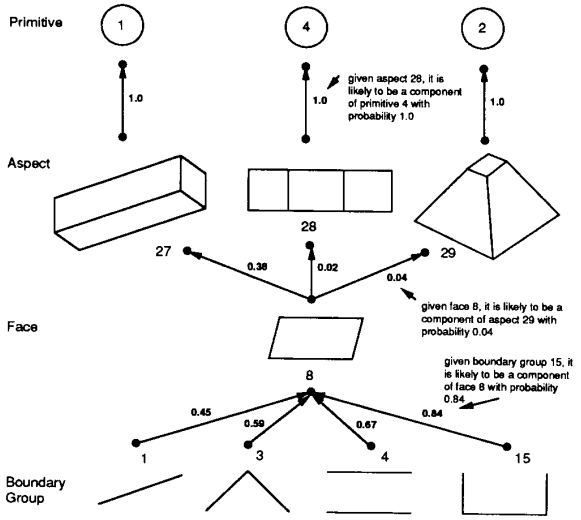

Fig. 7. Combining the object-centered and viewer-centered models.

probabilistic constraints on the boundary group to face, face to aspect, and aspect to primitive mappings. The histograms in Fig. 8 suggest that for 3-D modeling primitives that resemble the commonly used generalized cylinders [9], [1], [11], superquadrics [39], [40], or geons [8], [4], the most appropriate image features for recognition appear to be image regions or faces. Moreover, the utility of a face description can be improved by grouping the faces into the more complex aspects, thus obtaining a less ambiguous mapping to the primitives and further constraining their orientation. Only when a face's shape is altered due to primitive occlusion or intersection should we descend to analysis at the contour or boundary group level.

III. Primitive Recovery

Given an image of one or more objects, the goal of primitive recovery is to extract instances of the primitives and their connectivity relations. Our approach first segments the input image into regions and determines the possible face labels for each region. Next, we assign aspect labels to the faces, effectively grouping the faces into aspects. Finally, we map the aspects to primitives and extract primitive connectivity. The following sections describe our approach in detail through the aid of a comprehensive example.

A. Face Extraction

From the histograms in Fig. 8, we concluded that the aspect to primitive mapping was the least ambiguous mapping to the primitives. In addition, we concluded that the best mapping to the aspects was from the faces rather than from the boundary groups. This suggests that faces are an appropriate starting point in the primitive recovery process. Since we characterize faces by their bounding contours, our first step is to extract a set of contours from the image; this can be accomplished using either region-based or edge-based methods.[3]

Once a set of contours has been extracted from the image, the next step is to partition the contours at significant curvature discontinuities. The segmented contours are captured in a *contour graph* in which nodes represent junctions or significant curvature discontinuities, and arcs are the actual bounding face contours. Fig. 9 illustrates the enumerated contours of a scene containing an object composed of two blocks.

Given the contour graph representation of an input scene, our next task is to construct its corresponding *face graph*, in which nodes represent faces, and arcs represent face adjacencies.[4] We represent the contour graph as an adjacency matrix, where each edge appears twice; the edges incident to each node are ordered clockwise with respect to some reference.[5] The algorithm for transforming the contour graph into a face graph is based on [47] and proceeds as follows.

Algorithm for Producing the Face Graph: Start at any node n_s and follow a sequence of edges, deleting each edge in the sequence, until node n_s is again encountered; the edges in this cycle form a face. To ensure that only minimal cycles are followed, when arriving at node n_i from edge e_j, we must leave node n_i on the edge that is the clockwise neighbor of e_j. We repeat the process until all edges in node n_s's adjacency list have been deleted; any node with a nonempty adjacency list then becomes the new start node. When all edges in the graph have been deleted, the algorithm terminates. The final step involves the deletion of the "background face," i.e., the face representing the area external to the graph. This can be identified by keeping track of the internal angles of all faces and eliminating the face whose internal angle sum exceeds $180(n-2)$, where n is the number of segments bounding the face.[6]

Each edge can be a member of, at most, two faces. By keeping track of face membership when deleting an edge, we can keep track of face connectivity. Since each edge in the adjacency list is encountered exactly twice, the complexity of transforming a contour graph into a face graph is $O(e)$, where e is the number of edges in the contour graph. Fig. 10 illustrates the face graph generated from the contour graph in Fig. 9.

B. Face Labeling

Once the faces have been extracted, we must classify each face according to the faces in the aspect hierarchy. A face is represented by a contour graph, in which nodes represent the face's contours, and arcs represent relations between contours.[7] Thus, the classification of a face in the image consists of comparing its graph to those graphs representing the faces

[3] For our task of extracting faces, region-based methods are preferable since they avoid the problem of contour gaps, which can break the cycle of contours bounding a region.

[4] Two faces are said to be adjacent if they share one or more contours in the contour graph representation.

[5] If the segment is curved, then the tangent to the curve at the point of incidence is used to define its orientation.

[6] For the purpose of calculating internal angles, each curved segment is replaced by a piecewise linear approximation.

[7] Two adjacent collinear or curvilinear contours bounding a face may have been separated in the contour graph by a junction, e.g., contours 8 and 9 of the face consisting of the contours 8, 9, 10, 7, and 2 in Fig. 9. If so, they are merged to form one node in the graph. In addition, all nodes are classified as either a straight line, a concave curve, or a convex curve.

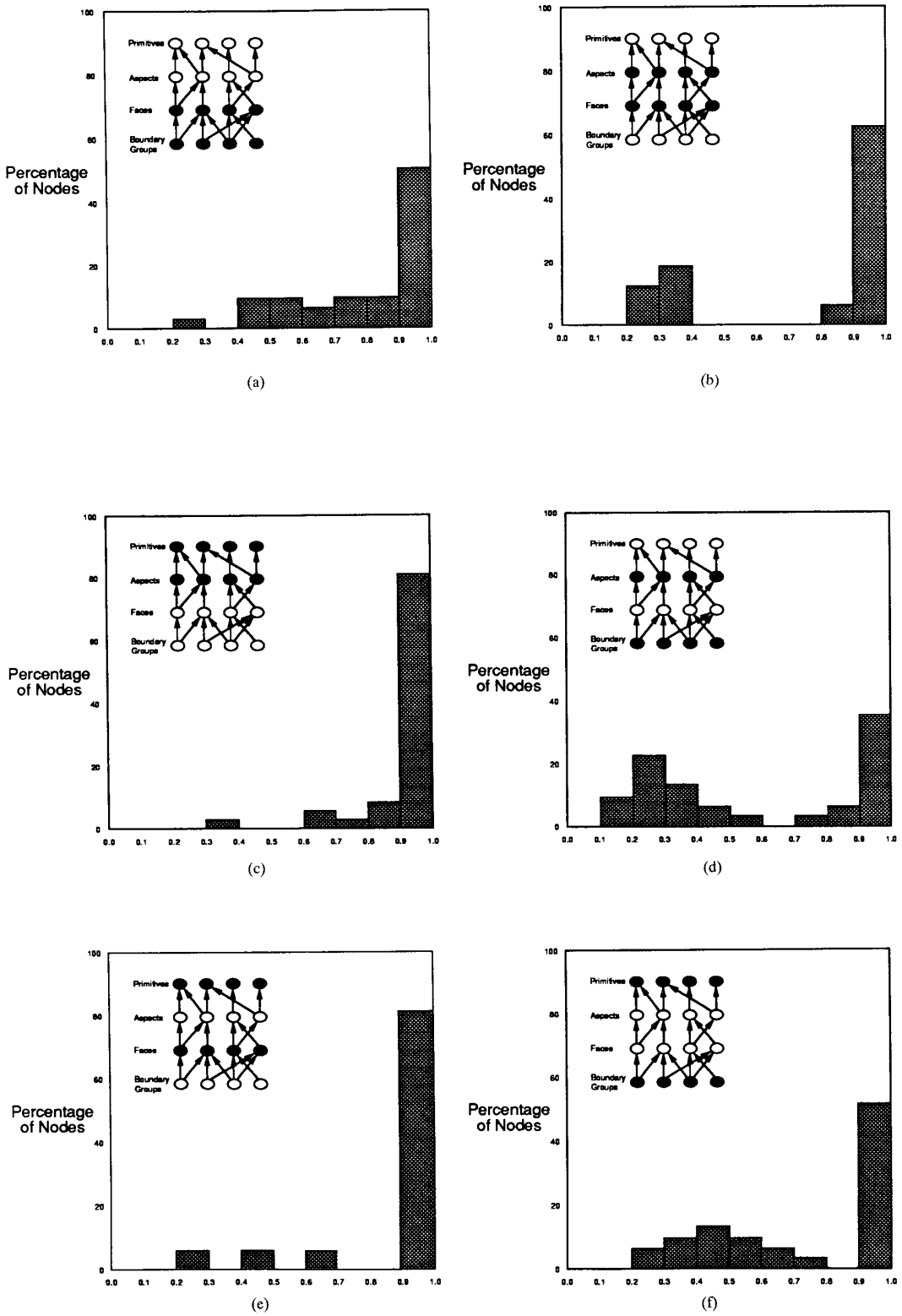

Fig. 8. Analysis of mappings between levels of the aspect hierarchy: (a) Probability of most likely face for each boundary group; (b) probability of most likely aspect for each face; (c) probability of most likely primitive for each aspect; (d) probability of most likely aspect for each boundary group; (e) probability of most likely primitive for each face; (f) probability of most likely primitive for each boundary group.

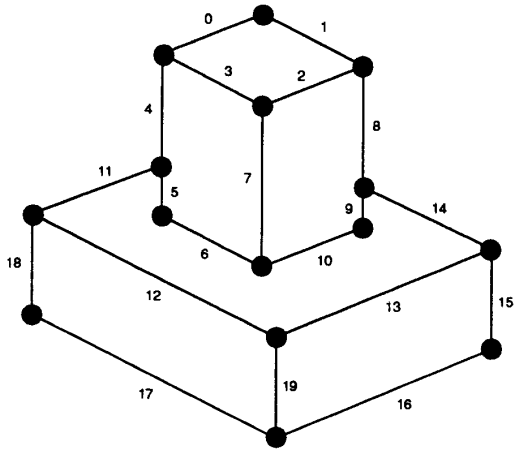

Fig. 9. Contour graph representation of an input scene.

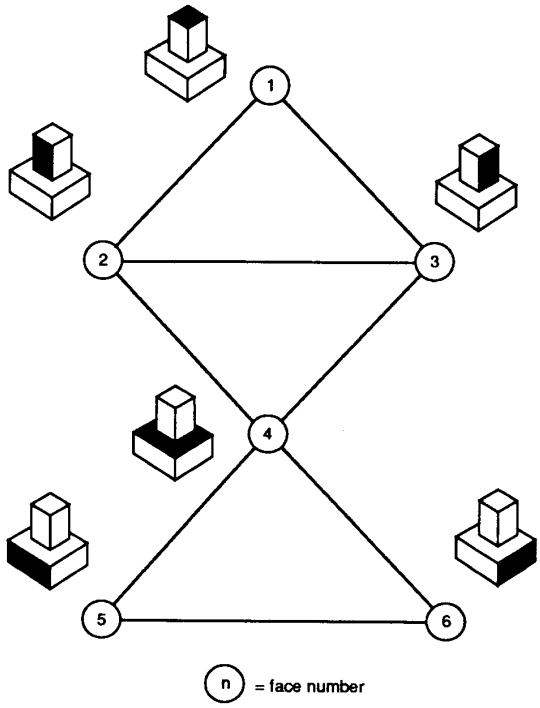

Fig. 10. Face graph corresponding to contour graph in Fig. 9.

in the aspect hierarchy. If there is an exact match, then we immediately generate a *face hypothesis* for that image face, identifying the label of the face. If, due to occlusion, there is no match, we must descend to the boundary group level of the aspect hierarchy. We then compare *subgraphs* of the graph representing the image face to those graphs at the boundary group level of the aspect hierarchy. For each subgraph that matches, we generate a face hypothesis with a probability determined by the appropriate entry in the conditional probability matrix mapping boundary groups to faces. The complexity of classifying an image face is polynomial in the number of contours in its corresponding graph since both the size and number of the graphs in the aspect hierarchy are fixed.[8]

Thus, from the original contour graph representation of the image, we first construct a face graph, and then for each face in the face graph, the classification process results in a list of hypotheses about the face's label. In the simple case of an image face that exactly matches a face found in the aspect hierarchy, the list contains a single hypothesis with probability 1.0.[9] For an image face that does not exactly match a face found in the aspect hierarchy, the list contains one or more face hypotheses listed in decreasing order of probability.[10] Each face hypothesis lists in decreasing order of probability its *seed contour sets*, i.e., contours that gave rise to the hypothesis. In the case of an image face that exactly matches a face in the aspect hierarchy, the face has a single seed contour set containing all of the face's bounding contours. On the other hand, a face hypothesis inferred from one or more boundary groups will have a seed contour set for each boundary group, containing the contours making up the boundary group.

As an example, Table I lists the face hypotheses, in decreasing order of probability, for each face in Fig. 10. The *face number* field refers to the face numbers in Fig. 10, the *face label* field gives the hypothesis label (Fig. 5), and the *probability* field gives the probability of the face hypothesis. The *seed contour sets* field gives the face hypothesis seed contour sets, each containing a set of numbered contours from Fig. 9 (with adjacent collinear or curvilinear contours parenthesized), while the *boundary group label* field gives the boundary group label of those seed contour sets corresponding to a boundary group.

C. Extracting Aspects

1) Problem Definition: We now have a face graph with one or more face hypotheses at each face. We can formulate the problem of extracting aspects as follows: Given a face graph and a set of face hypotheses at each face, find a covering of the face graph using aspects in the aspect hierarchy (an *aspect covering*) such that no face is left uncovered, and each face is covered by only one aspect, or, more formally, given an input face graph FG, partition the vertices (faces) of FG into disjoint sets $S_1, S_2, S_3, \ldots, S_k$ such that the graph induced by each set S_i is isomorphic to the graph representing some aspect A_j from a fixed set of aspects $A_1, A_2, A_3, \ldots, A_n$.

There is no known polynomial time algorithm to solve this problem (see Appendix A); however, the conditional

[8] If the maximum number of contours in an aspect hierarchy face is k and the number of contours in the image face graph is n, then the complexity of classifying the face is $O(n^k)$ for the given aspect hierarchy.

[9] Due to occlusion, the fact that an image face exactly matches an aspect hierarchy face does not guarantee that the interpretation (label) of the image face is correct. A more precise analysis would go ahead and compare the image face's boundary groups to aspect hierarchy boundary groups, ensuring that the correct face hypothesis is generated. Nevertheless, the hypothesis representing the matched face would still have the highest probability.

[10] There can be no redundancy in this list since we do not allow multiple face hypotheses with the same label. If two boundary groups give rise to the same face hypothesis, the hypothesis is assigned the higher probability of the two boundary group to face mappings.

TABLE I
FACE HYPOTHESES FOR FACE GRAPH IN FIG. 10

Face Number	Face Label	Probability	Seed Contour Sets	Boundary Group Label
1	8	1.00	(3 2 1 0)	
2	8	1.00	(6 7 3 (4 5))	
3	8	1.00	(2 7 10 (9 8))	
4	8	0.84	(11 12 13)	15
			(12 13 14)	15
	9	0.16	(11 12 13)	15
			(12 13 14)	15
5	8	1.00	(18 17 19 12)	
6	8	1.00	(19 16 15 13)	

TABLE II
ASPECT HYPOTHESES FOR FACES 0 AND 3 OF FACE GRAPH IN FIG. 10

Face Number	Aspect Label	Aspect Prob.
0	27	0.38
	19	0.14
	4	0.10
	29	0.04
	35	0.04
	34	0.04
	31	0.04
	18	0.04
	30	0.04
	20	0.04
	36	0.03
	25	0.02
	33	0.02
	37	0.02
	28	0.02
	26	0.01
3	27	0.32
	19	0.12
	4	0.09
	29	0.04
	29	0.04
	37	0.03
	35	0.03
	34	0.03
	31	0.03
	18	0.03
	30	0.03
	20	0.03
	30	0.03
	33	0.03
	36	0.03
	25	0.02
	33	0.02
	37	0.02
	20	0.01
	28	0.01
	21	0.01
	5	0.01
	26	0.00

probability matrices provide a powerful constraint that can make the problem tractable. After the previous steps, each face in the face graph has a number of associated face hypotheses. For each face hypothesis, we can use the face-to-aspect mapping to generate the possible *aspect hypotheses* that might encompass that face; the face hypothesis becomes the *seed face hypothesis* of each of the resulting aspect hypotheses. The probability of an aspect hypothesis is the product of the face-to-aspect mapping and the probability of its seed face hypothesis. At each face, we collect all the aspect hypotheses (corresponding to all face hypotheses) and rank them in decreasing order of probability. For faces 0 and 3 in Fig. 10, Table II lists the aspect hypotheses in decreasing order of probability (faces 1, 2, 4, and 5 are equivalent to face 0).

Each aspect hypothesis is merely an informed guess as to the aspect label of its seed face hypothesis. The process of verifying the hypothesized aspect label is called *aspect instantiation*. For an aspect to be instantiated from an aspect hypothesis, the relations between the seed face hypothesis and neighboring face hypotheses must be consistent with the definition of the aspect. More formally, there must exist a set of faces S including the face corresponding to the seed face hypothesis, such that the face subgraph induced by S is isomorphic to the graph representing the aspect. Since there may be multiple sets of faces that satisfy this criteria, there may be multiple aspects instantiated from a single aspect hypothesis. Hence, the process of aspect instantiation produces a (possibly empty) set of instantiated aspects for a given aspect hypothesis. For a detailed discussion of aspect instantiation and how occluded aspects can be instantiated, see Appendix B.

We can now reformulate our problem as a search through the space of aspect labelings of the faces in our face graph. In other words, we wish to choose one aspect hypothesis from the list at each face such that the instantiated aspects completely cover the face graph. There may be many labelings that satisfy this constraint. Since we cannot guarantee that a given aspect covering represents a correct interpretation of the scene, we must be able to enumerate, in decreasing order of likelihood, all aspect coverings until the objects in the scene are recognized.

2) Algorithm for Enumerating Aspect Coverings: For our search through the possible aspect labelings of the face graph, we employ Algorithm A [38] with a heuristic based on the probabilities of the aspect hypotheses. The different labelings are ordered in the open list according to a value determined by the heuristic function. At each iteration, a labeling, or state, is removed from the open list and checked to see if it represents a solution (a covering). The successor states are then generated, evaluated, and added to the open list. The actual instantiation of aspects is performed during successor generation. The algorithm continues until all possible solutions are found, i.e. all labelings are checked. However, it should be pointed out that in an object recognition framework, once a solution is found, the search would only be continued if the recovered shapes (inferred from the aspect covering) could not be recognized. The following sections describe the components of the algorithm in more detail.

a) State Definition: A state has two components: a selection of instantiated aspects and a numeric value determined by the heuristic function. The selection of instantiated aspects is actually a list of indices (one per face) with each index pointing to a set of one or more aspects instantiated from a particular aspect hypothesis. For a given face, the value of the index

```
input: node
output:
value value = 0
for each index in node do
     aspect set = set of aspects pointed to by index
     aspect hypothesis = aspect hypothesis from which aspect set was instantiated
     value = value
          - probability(aspect hypothesis) -
            (c1 * maximum number of visible faces of any aspect in aspect set
                  / number of faces in aspect hypothesis definition) -
            (c2 * maximum number of visible faces of any aspect in aspect set)
return value
```

Fig. 11. Heuristic for evaluating a state.

ranges from one to the number of aspect hypotheses associated with that face. Therefore, since the aspect hypotheses are ranked in decreasing order of probability, a lower index points to a set of aspects instantiated from a higher probability aspect hypothesis. The numeric value is used to order the states in the open list by increasing value; a more promising state has a lower value than a less promising state and appears sooner in the open list.

b) Generating Successor States: To generate a successor state, i.e., a new selection of instantiated aspects, we increment one of the indices of the parent state to point to the next nonempty list of instantiated aspects. Consider, for example, a face graph consisting of three faces, and suppose we want to generate the successors of the state having the index set $(1, 1, 1)$. Consider the index corresponding to the first face. First, we increment the index to point to the set of aspects instantiated from the second most likely aspect hypotheses for the first face. If we have not yet attempted to instantiate a list of aspects from that hypothesis, we do so, producing a possibly empty set of instantiated aspects. If the set of instantiated aspects is empty, we increment the index and repeat the process until the next nonempty set is found. The resulting index is combined with the original indices corresponding to the second and third faces to form the new state. Therefore, the first successor is $(n_1, 1, 1)$, where n_1 is the index pointing to the next most likely nonempty list of instantiated aspects corresponding to the first face. The process is repeated for each index (face), yielding a total of three successors: $(n_1, 1, 1)$, $(1, n_2, 1)$, and $(1, 1, n_3)$. If, for a given successor, the index is incremented beyond the number of aspect hypotheses for that face, the successor is discarded. Before each successor state is added to the open list, it is evaluated by the heuristic function. Provided the successor state does not already appear in either the open or closed lists, it is merged into the open list and sorted by increasing value.

c) Verifying a Solution State: When a state is removed from the open list, we check to see if it represents a solution state. The state is then added to the closed list, provided that it does not already appear on the closed list. For each index in the state's index list, we gather the corresponding set of instantiated aspects. If the state is a solution state, there must be a selection of instantiated aspects (one per set) such that together, they completely cover the face graph. Since we wish to enumerate all the solutions, we must return all selections that cover the face graph. To generate the selections, we perform a depth-first search. Note that for an aspect with label t to cover faces $f_1, f_2, \ldots, f_m$, there must exist an aspect with label t covering faces $f_1, f_2, \ldots, f_m$ in each of the sets of instantiated aspects corresponding to faces $f_1, f_2, \ldots, f_m$, respectively. A solution is added to the solution list only if it does not already appear in the solution list.

d) Evaluating a State: Before adding a successor state to the open list, it is evaluated using a heuristic function. The heuristic has been designed to meet three objectives. First, we favor selections of aspects instantiated from higher probability aspect hypotheses. Second, we favor selections whose aspects have fewer occluded faces since we are more sure of their labels. Finally, we favor those aspects covering more faces in the image; we seek the minimal aspect covering of the face graph. These three objectives have been combined to form the algorithm for evaluating a state, which is shown in Fig. 11. For the experiments described in Section V, the values of c_1 and c_2 were empirically chosen to be 0.25 and 0.50, respectively.

3) Example: We have applied the algorithm for enumerating the aspect coverings of a face graph to the face graph in Fig. 10; the first 12 coverings are shown in Fig. 12. For each covering, we indicate the partitioning of the faces into aspects (shading), the labeling of the aspects, and the number of states removed from the open list (i.e., iterations of the algorithm) when the covering was found. For this simple example, the first expanded state represents the correct covering.

D. Extracting Primitives

We can represent an aspect covering by a graph in which nodes represent aspects and arcs represent aspect adjacencies. For two adjacent aspects A and B, the arc labels consist of one or more pairs of indices that represent face adjacencies. For example, arc label (i, j) indicates that face i in the graph corresponding to A is adjacent to face j in the graph corresponding to B. Fig. 13 illustrates the graph representing the first aspect covering enumerated in Fig. 12. Given the graph representing an aspect covering, the next steps are to map the aspects in the covering to a set of primitives and to extract their connectivity. The following sections describe this process in detail.

1) Algorithm for Enumerating Primitive Coverings: For each aspect in the aspect covering, we can use the aspect to

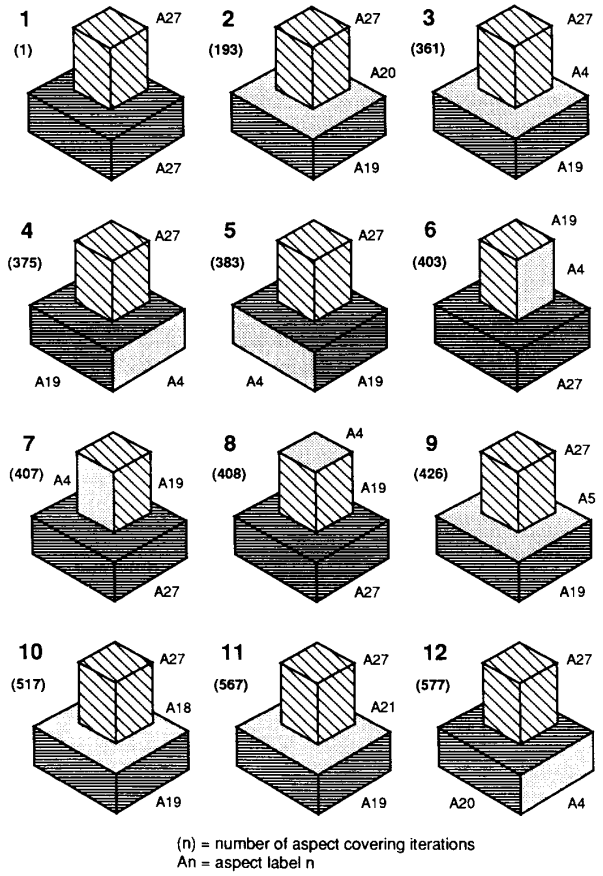

Fig. 12. Enumerated aspect coverings for face graph in Fig. 10.

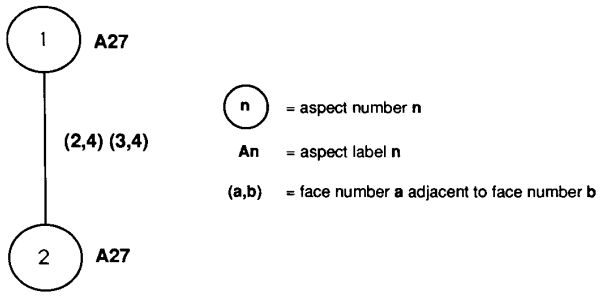

Fig. 13. Graph representing first aspect covering in Fig. 12.

primitive mapping to hypothesize a set of *primitives*; the aspect becomes the *seed aspect* of each of the resulting primitives.[11] Table III lists the primitives hypothesized for each aspect in Fig. 13; as in the case of aspect hypotheses generated from face hypotheses, we can rank the primitives in decreasing order of probability. A selection of primitives (one per aspect) represents a 3-D interpretation of the aspect graph; we call such a selection a *primitive covering*. Since we cannot guarantee that a given primitive covering represents a correct interpretation of the scene, we must be able to enumerate, in decreasing order of likelihood, all primitive coverings until the objects in the scene are recognized.

[11] In addition, the aspect hierarchy defines a mapping from the faces in an aspect to the attachment surfaces of a primitive.

TABLE III
PRIMITIVE HYPOTHESES FOR ASPECT COVERING IN FIG. 13

Aspect Number	Primitive Label	Primitive Prob.	Seed Aspect Label	Face to Surface Map
0	1	1.0	27	0 1 NA NA 2 NA
1	1	1.0	27	3 4 NA NA 5 NA

To enumerate the selections, we employ a variation on the search algorithm used to enumerate the aspect coverings. Again, a state consists of a list of indices (one per aspect) and a value used to order the states in the open list; the index corresponding to an aspect points to a primitive hypothesized from that aspect. Since one aspect's hypothesized primitives are independent of another's, no verification of nodes removed from the open list is necessary. Successor generation is the same as that used in the aspect covering algorithm; however, since no primitive instantiation is necessary, an index need only be incremented by 1. The heuristic evaluation function simply negates the sum of the probabilities of the primitives, thereby favoring higher probability interpretations.

2) Extracting Primitive Connectivity: A primitive covering, which is represented by a graph in which nodes represent primitives and arcs represent primitive adjacencies, is then compared with the object database during the recognition process. If two aspects are not adjacent in the aspect covering, their corresponding primitives are not adjacent in the primitive covering. However, if two aspects are adjacent in the aspect covering, this does not mean that their corresponding primitives are necessarily adjacent in 3-D; one primitive may be occluding the other without being attached to it. A primitive connection between primitives P_1 and P_2 is said to be visible if the following condition is satisfied:

- There exists a pair of faces F_1 and F_2 such that F_1 belongs to the aspect corresponding to P_1 and F_2 belongs to the aspect corresponding to P_2, F_1 and F_2 are adjacent in the face graph, and F_1 and F_2 share a contour (following collinearity or curvilinearity grouping).[12]

Therefore, we define two types of primitive connectivity based on connection visibility:

- Two primitives are said to be *strongly* connected if their corresponding aspects are adjacent in the aspect graph, and the primitive connection is visible; in this case, we assume that the primitives are attached.
- Two primitives are said to be *weakly* connected if their corresponding aspects are adjacent in the aspect graph, and the primitive connection is *not* visible; in this case, one primitive occludes the other, and it is not known whether or not they are attached.

[12] Before grouping, two adjacent faces in the face graph share a contour by definition. However, following collinearity and curvilinearity grouping within their respective faces, they may not have a contour in common.

A strong primitive connection strongly suggests the existence of a connection between two primitives. We can enhance the indexing power of a strongly connected subgraph if the attachment surfaces (Fig. 2) involved in each connection are hypothesized. Although it is impossible to define a set of domain independent rules that will, for any given set of primitives, correctly specify the attachment surfaces involved in a connection, we can define a set of heuristics that will specify a set of likely candidates. If a strongly connected subgraph is common to two object models, these heuristics can be used to rank order the candidates for verification.

Hypothesizing the attachment surfaces proceeds as follows. Let S_1 be the set of faces belonging to the aspect corresponding to P_1, which are adjacent to a face belonging to the aspect corresponding to P_2. Similarly, let S_2 be the set of faces belonging to the aspect corresponding to P_2, which are adjacent to a face belonging to the aspect corresponding to P_1. There are three cases to consider:

1) *Sets S_1 and S_2 each contain a single face.* The attachment surface for P_1 is among the set of attachment surfaces that are adjacent to, and including, the surface representing the face in S_1. The attachment surface for P_2 is among the set of attachment surfaces that are adjacent to, and including, the surface representing the face in S_2. More intuitively, we believe that the attachment surface is in the local vicinity (on the primitive) of the attachment surface corresponding to the single visible face.
2) *Set S_1 contains a single face, and set S_2 contains multiple faces.* In this case, the attachment surface for P_1 is the surface that the face in S_1 maps to. The attachment surface for P_2 is among the set of surfaces that are adjacent to, but not included in, the surfaces representing the faces in S_2. More intuitively, we believe that P_2 penetrates P_1; since the connection is visible, the attachment surface for S_1 is therefore attached to an occluded surface of P_2. (The same holds true when set S_2 contains a single face and set S_1 contains multiple faces.)
3) *Sets S_1 and S_2 both contain multiple faces.* In this case, the attachment surface for P_1 is among the set of surfaces that are adjacent to, but not included in, the set of surfaces representing the faces in S_1. The attachment surface for P_2 is among the set of surfaces that are adjacent to, but not included in, the set of surfaces representing the faces in S_2. More intuitively, we believe that although the attachment of P_1 and P_2 is visible, both their attachment surfaces are occluded.

Although the scope of this paper is the recovery and not recognition of objects, it is important to note that it is at the primitive covering step that the object model database is brought into play. Given a primitive covering with strong and weak connections (arcs), we have developed a recognition strategy whereby strongly connected subgraphs are used to index into the object database with verification addressing the weak connections. If a portion of the graph matches an object, then subsequent primitive or aspect covering enumeration can be focused on unrecognized portions of the image. In this manner, the search for both aspect and primitive coverings can be constrained by the object model database. A description of this recognition strategy, along with preliminary results, can be found in [17].

IV. Coping with Segmentation Errors

Until now, the discussion has assumed not only a correct region segmentation of the input image but also a correct partitioning of the contours bounding the regions. However, this assumption will often not be valid. Within the context of our system, two types of segmentation errors are possible. In the first case, the image regions are correctly segmented, but their bounding contours are incorrectly segmented or classified. In the second case, the image regions are incorrectly segmented. The following sections discuss our approach to dealing with these two problems.

A. Contour Segmentation Errors

Although an image region may be correctly segmented, there may be errors in the graph representing its bounding contours. For example, the partitioning of the contours may be incorrect, a given contour may be misclassified as straight or curved, or a contour relation (e.g., parallelism) may be incorrect. Even if we ignore these errors, a correct interpretation of the scene may still be possible. If there exists a subgraph that is correctly labeled, i.e., a correctly labeled boundary group, the correct face hypothesis will be generated. Of course, if the subgraph is smaller, a greater number of different face hypotheses will be generated, thereby increasing the space of labelings that must be examined during the aspect covering process.[13]

A more effective approach is to associate probabilities with the labeling of each contour (node) and contour relation (arc) in the graph [33]. Although this would result in a greater number of face hypotheses for an image face, the face labeling would be less sensitive to errors in contour classification and perceptual grouping. However, this still assumes that the nodes in the graph are correct, i.e., the contours bounding the face have been correctly partitioned.

Our approach to the curve partitioning problem is to assume an *oversegmentation* of the bounding contour and use the aspect hierarchy to perform a model-based merging of adjacent contours during the face labeling stage. An oversegmentation can normally be obtained by using conservative parameters in most curve partitioning algorithms, e.g., [46], [32], [21].

If an image face matches an aspect hierarchy face, we assume that the image face is correct. However, if no match is possible, recall that we attempt to match subgraphs of the graph representing the image face to boundary groups in the aspect hierarchy. Let F be the graph representing the image face, and let S be a subgraph of F that matches some boundary group B in the aspect hierarchy. Consider a contour c_f such that c_f is a member of F, c_f is not a member of S, and c_f is adjacent to a contour c_s in S. We replace c_s with the merger

[13] The face hypotheses inferred from a smaller subgraph (boundary group) are likely to have lower probabilities.

of c_f and c_s, provided that c_f and c_s are similar according to some criteria, and examine the resulting S.[14] If the new S still matches B, we retain the merge; otherwise, we discard it. The process continues until no new merges are possible. If, during the merging procedure, S becomes closed, then we match S to faces in the aspect hierarchy.

B. Region Segmentation Errors

The other type of error we must address is an incorrect region segmentation. Our approach to the region segmentation problem is very similar to our approach to the contour segmentation problem. Assume an oversegmentation of the image regions, and use the aspect hierarchy to perform a model-based merging of adjacent regions during the aspect labeling stage. An oversegmentation can normally be obtained by using conservative parameters in most region segmentation algorithms (see [26] for a review of region segmentation algorithms).

Consider a region in the input image such that, due to oversegmentation, the region is split into two smaller regions. Let the graph representing the complete region be F, and let those representing the two component regions be f_i and f_j. Since each component region is a subset of the complete region, f_i and f_j must each have at least one subgraph in common with F. Therefore, if the correct label of F is l, then f_i and f_j must have, in their respective ranked lists of face hypotheses, a face hypothesis with label l. During the aspect instantiation phase, a group of neighboring faces, including face f_i, is checked to see whether the faces satisfy the definition of a particular aspect. If face f_i is supposed to contain (among its face hypotheses) a face hypothesis with label l, we first examine the faces neighboring face f_i. Since one of these neighboring faces f_j has (among its face hypotheses) a hypothesis with the correct label, faces f_i and f_j are candidates for a merge. More specifically, let fh_i be the face hypothesis with label l belonging to f_i. If the face resulting from the merging of faces f_i and f_j gives rise to a face hypothesis with label l having a probability greater than the probability of fh_i, then the merge is retained. If the two faces are merged, we repeat the process with the new face, terminating when no merges are performed in a given iteration.

Fig. 14 illustrates how this technique is applied to a scene in which region oversegmentation has occurred. In this case, one of the faces of the upright block has been oversegmented to yield two faces, numbered 1 and 2. The face labels and, if applicable, the boundary group labels used to generate the face labels, are given in the table. Consider the iteration of the aspect enumeration algorithm that checks the labeling in which faces 0, 1, and 3 are each represented by the aspect label 27. For this aspect to be instantiated, each of faces 0, 1, and 3 must have (among their possible face labels) the face label 8. Since face 1 has the correct face label, we invoke the region merging algorithm. Face 1 has three neighboring faces excluding 0 and 3; therefore, we examine the results of three merges: faces 1 and 2, 1 and 6, and 1 and 4. The only merge yielding a face whose resulting label is 8 and whose probability (1.0) exceeds that of face 1 is the merge of faces 1 and 2. Hence, this merge is retained, and the oversegmentation is rectified.

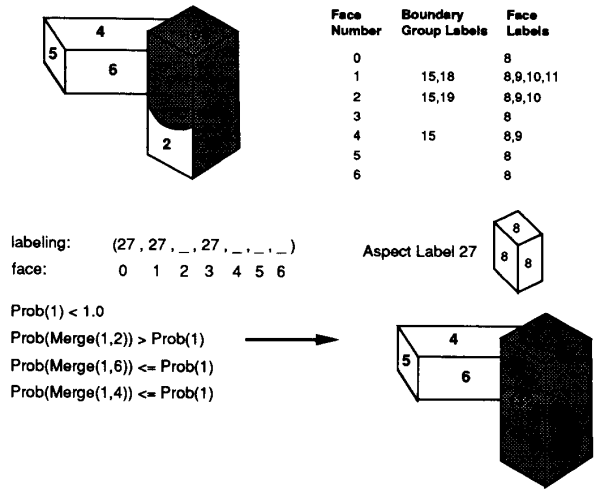

Fig. 14. Rectification of region oversegmentation.

The above model-based region merging algorithm is based on the probabilities inherent in the aspect hierarchy. Given a label specification for an image region, we perform a constrained growth on that region while the probability of that region increases and terminate when the probability decreases. This algorithm, in conjunction with the proposed solutions to contour segmentation errors, enhances our primitive recovery algorithm by making it less sensitive to image segmentation performance.

V. Results

We have built a software system to demonstrate our approach to shape recovery; the system is known as the **O**bject recognition using **P**robabilistic **T**hree-dimensional **I**nterpretation of **C**omponent **A**spects (OPTICA) system and has been implemented on a Symbolics™ 3600. For the first set of examples, the image segmentation and perceptual grouping tasks are performed manually. Therefore, the input to the system is a manually segmented contour image; graphs representing all faces and boundary groups are entered manually. We are currently developing a front end to the system to accept real images. Although still at an early stage of development, we have successfully applied a complete working system to real images containing objects; the second set of examples reports these results.

A. Synthetic Data

For the first example, we apply the primitive recovery algorithm to our familiar scene containing the two blocks; the resulting primitive covering is presented in the large box to the left of Fig. 15. Each face in the image is described by a small box containing some mnemonics. The mnemonics PN, PL, PP,

[14] An example criterion for merging would be that both c_f and c_s are concave curves that meet at a discontinuity whose magnitude does not exceed some threshold.

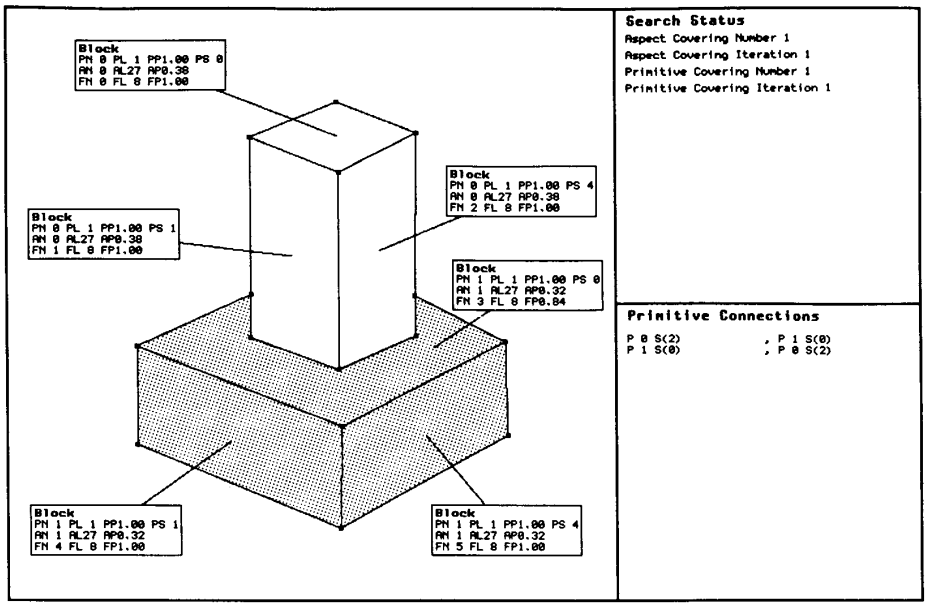

Fig. 15. Correct first interpretation of an object composed of two blocks.

and PS, refer to the primitive number (simply an enumeration of the primitives in the covering), primitive label (see Fig. 1), primitive probability, and primitive attachment surface (see Fig. 2), respectively, of the primitive that includes that face; the name of the primitive is given above the mnemonics. The mnemonics AN, AL, and AP refer to the aspect number (an enumeration), aspect label (see Fig. 6), and aspect probability, respectively, of the aspect from which the primitive was inferred; faces belonging to the same aspect (primitive) are shaded similarly. The mnemonics FN, FL, and FP refer to the face number (an enumeration), face label (see Fig. 5), and face probability, respectively, of the face from which the aspect was inferred. The smaller box to the upper right indicates the aspect covering iteration and primitive covering iteration (given the aspect covering); in this case, both the first aspect covering and primitive covering are correct. The smaller box to the lower right indicates the primitive connections by PN; for example, primitive PN0 is attached to primitive PN1. If two primitives are strongly connected, a list of probable attachment surfaces appears in parentheses next to the PN; for example, the connection between primitive PN0 and PN1 involves surface PS2 on primitive PN0 and surface PS0 on primitive PN1. Note that this list is not exclusive but, rather, a list of likely candidates. The time taken to generate the correct covering was approximately 8 s (including the time taken to read the input from disk).

In the second example, we apply the recovery technique to two views of a simple object consisting of a block attached to the side of a cylinder; the results are shown in Figs. 16 and 17. In each case, the first aspect and primitive coverings represent the correct interpretation of the scene. In Fig. 16, the algorithm converges quickly (10 s) since each primitive projects to a high probability aspect. In Fig. 17, however, the block primitive does not not project to its highest probability aspect. Consequently, more aspect hypotheses must be examined before the aspect covering is found (12 s).

In the third example, we apply the recovery technique to three views of a cooking pot. In Figs. 18 and 19, the algorithm quickly converges to the correct solution (7 and 5 s, respectively). In each case, the first aspect and primitive coverings represent the correct interpretation of the scene. However, in Fig. 20 (5 s), the first aspect covering and, consequently, the first primitive covering are incorrect. The pot body received the correct aspect label (AL 5) and primitive label (PL 6), but the pot handle did not (AL 19, PL 1).

The highest probability aspect that could be instantiated for the region corresponding to the pot handle was the aspect labeled 19. Consequently, the most likely primitive given aspect label 19 was the block primitive (PL 1). The correct interpretation, shown in Fig. 21 (6 s), represents the second aspect covering (two iterations) and the fifth primitive covering (given the correct aspect covering). This example demonstrates that ambiguous mappings between the faces and aspects, and between the aspects and primitives, result in ambiguous interpretations of the scene. This is where the object model database can be used to constrain the search. For example, if the first primitive covering (Fig. 20) was used as an index into the object database, the correct interpretation of the pot body could be used to constrain the search for an interpretation of the handle.

The next example using synthetic data demonstrates the algorithm applied to the desk lamp drawing from Bergevin and Levine [4]; the results are presented in Fig. 22. The first aspect and primitive coverings represent the correct interpretation of the scene (5 s). In Fig. 23, the algorithm is applied to an occluded scene containing the desk lamp and the cooking

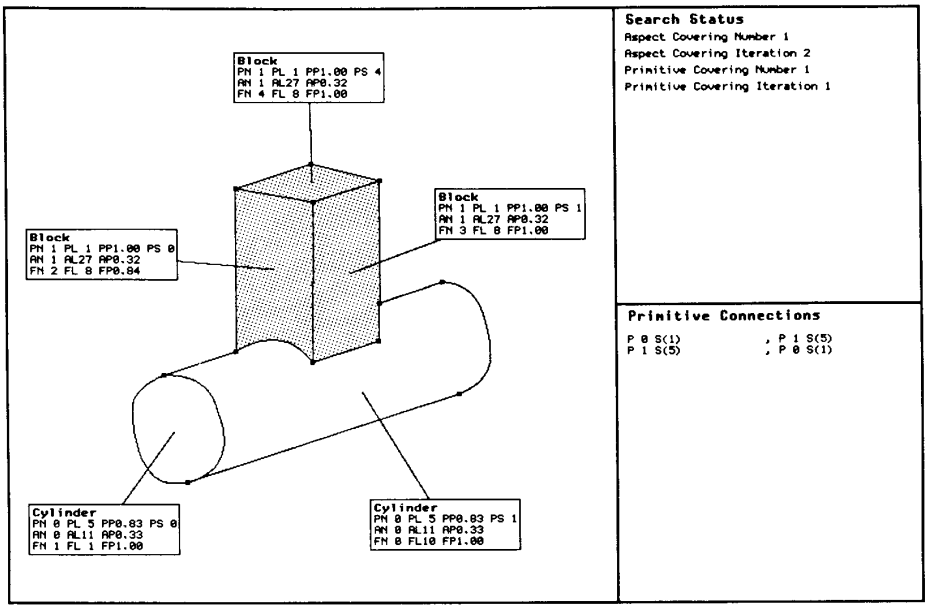

Fig. 16. Correct first interpretation of an object composed of a block attached to a cylinder (first view).

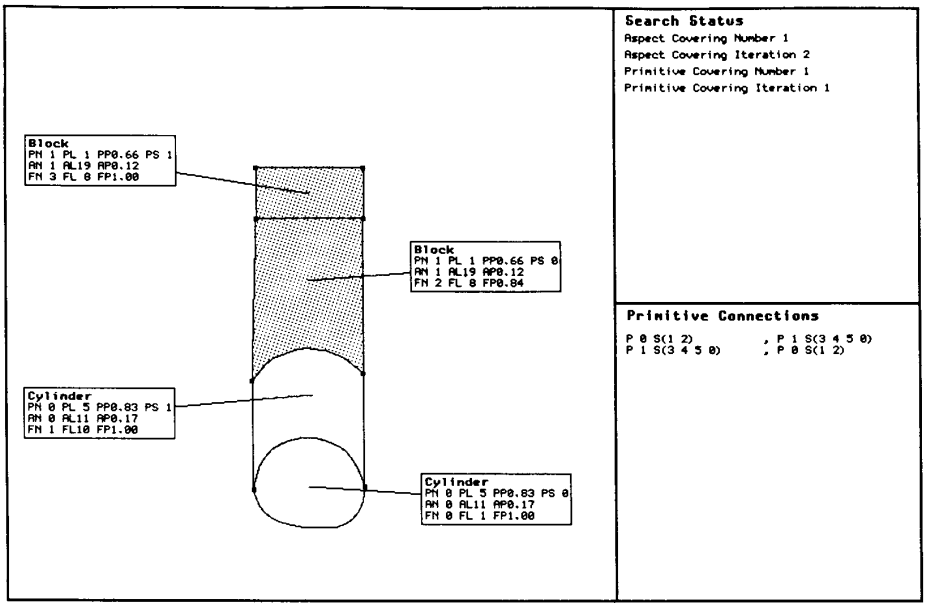

Fig. 17. Correct first interpretation of an object composed of a block attached to a cylinder (second view).

pot. Again, the first aspect and primitive coverings represent the correct interpretation of the scene (7 s). Note that the handle of the cooking pot has been interpretated as two distinct primitives; no collinearity grouping is performed at the primitive level.

B. Image Data

The first step in fully automating our system is the segmentation of the image into homogeneous regions. To extract regions, we first apply the Canny edge detector [12] to the input image to detect the projected surface discontinuities of the object. From the resulting edge pixels, our goal is to locate cycles of edge pixels that bound regions. However, since there may be small gaps in the edge pixel map that break cycles, we dilate the edge image to fill the gaps. The result is then skeletonized to yield an image containing single pixel width contours. The entire sequence of steps is executed on a Sun™ 3 in approximately 140 s.

From the image of contours, we apply a connected components algorithm to extract a set of contours, where each

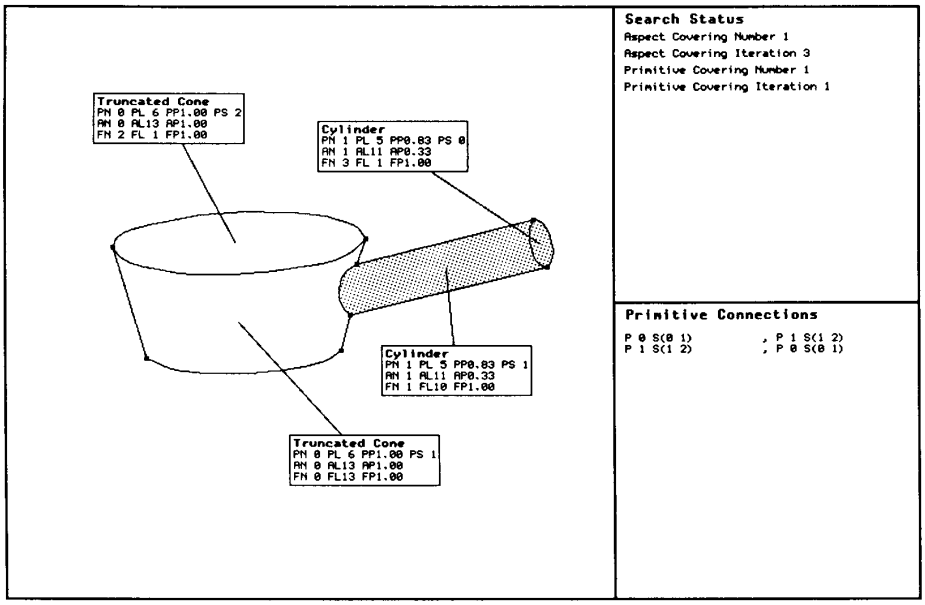

Fig. 18. Correct first interpretation of a pot (first view).

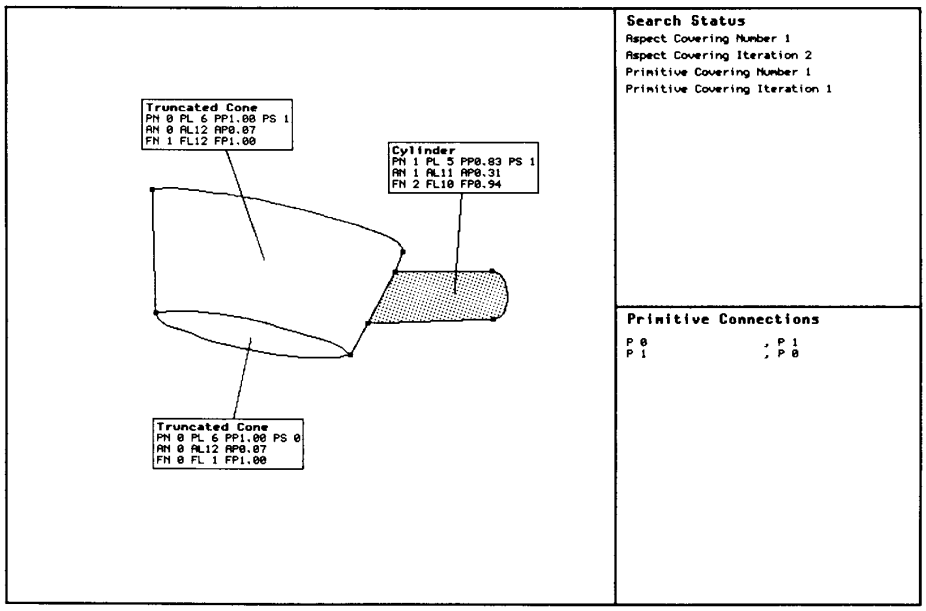

Fig. 19. Correct first interpretation of a pot (second view).

begins and ends at a junction of three or more contours; all other contours are discarded. The next step is to partition the contours at significant curvature discontinuities. We first apply Ramer's [46] algorithm to produce an initial set of breakpoints. Although effective for the partitioning of straight lines, the algorithm overpartitions curves. However, the resulting partition points are a superset of the correct partition points [32]. To remove the false partition points, we fit circular arcs to the left and right neighborhoods of each potential breakpoint and discard the breakpoint if the angle between the tangents to the two circles at the breakpoint is near 180°. The resulting set of contours are classified as lines or curves depending on how well a line can be fitted to them.

From the set of partitioned contours, we apply the algorithm in Section III-A to yield a face graph. For each face in the face graph, our next task is to represent the face by a graph in which nodes represent contours and arcs represent certain nonaccidental relations among the contours. For a given face, adjacent lines or adjacent curves that meet at a junction are merged (according to the criteria used to check initial

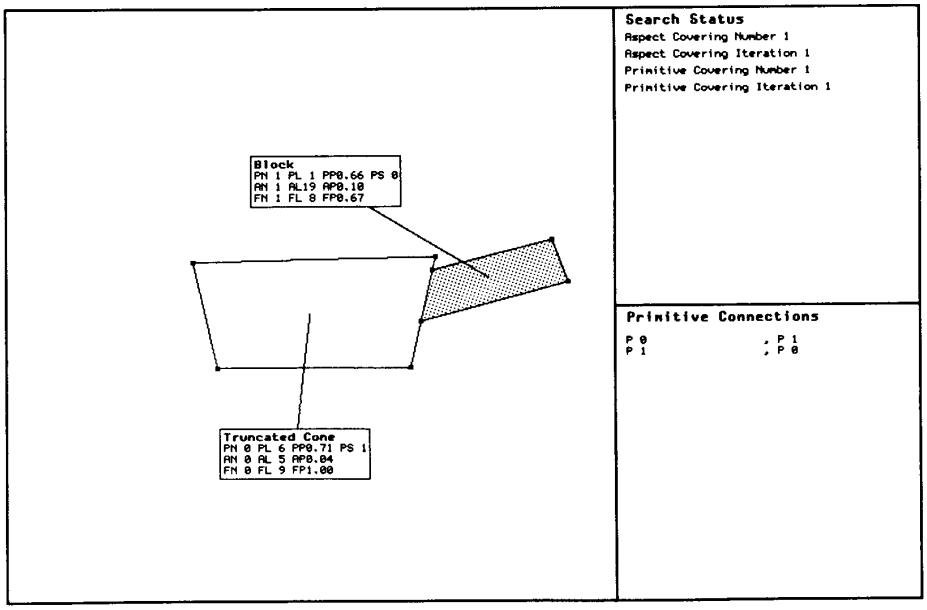

Fig. 20. Incorrect first interpretation of a pot (third view).

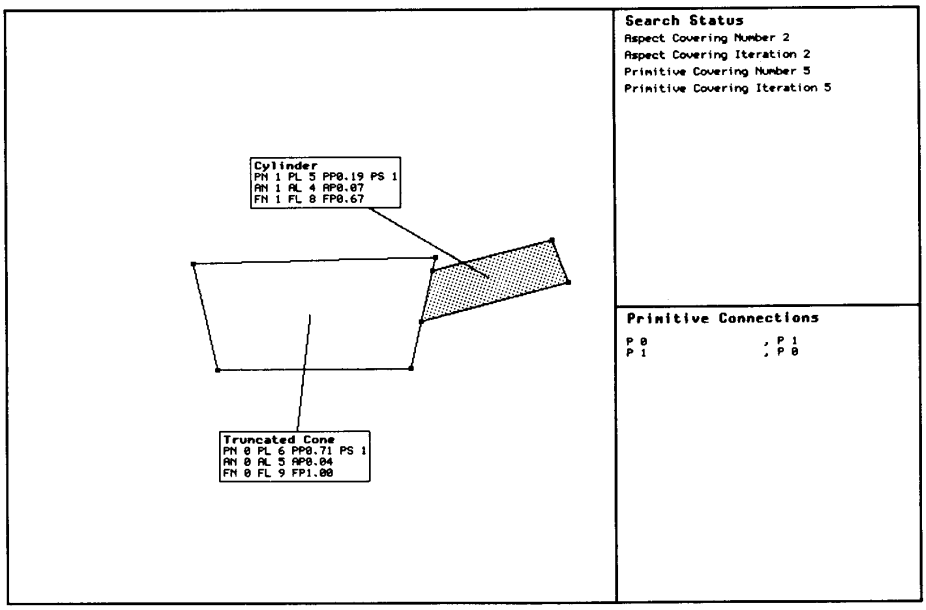

Fig. 21. Correct interpretation of a pot (third view).

partition points) if they are collinear or curvilinear. Any curves bounding the face are further classified as concave or convex by checking the angle between the lines joining the midpoint to the two ends of the curve. Nonadjacent lines are labeled parallel if the angle between them is small and symmetric if they are opposite and nonparallel. Nonadjacent curves are labeled parallel if one is concave, the other is convex, and they face in similar directions, where the direction of a curve is defined by the vector whose head is at the midpoint of the line joining the two ends of the curve and whose tail is that point on the curve whose perpendicular distance to the line is maximum. A similar test is used for curve symmetry if both curves are concave or convex. If, for parallel or symmetric curves, the radii of the circles fitted to the curves are significantly different, the relative size of the curves is noted.

The entire process was first applied to the 256×256 image of a table lamp as shown in Fig. 24. Excluding the time required to transform the raw image into a skeleton image (140 s), the time required to generate the first primitive covering was 60 s. Fig. 25 shows the partitioned contours extracted from the

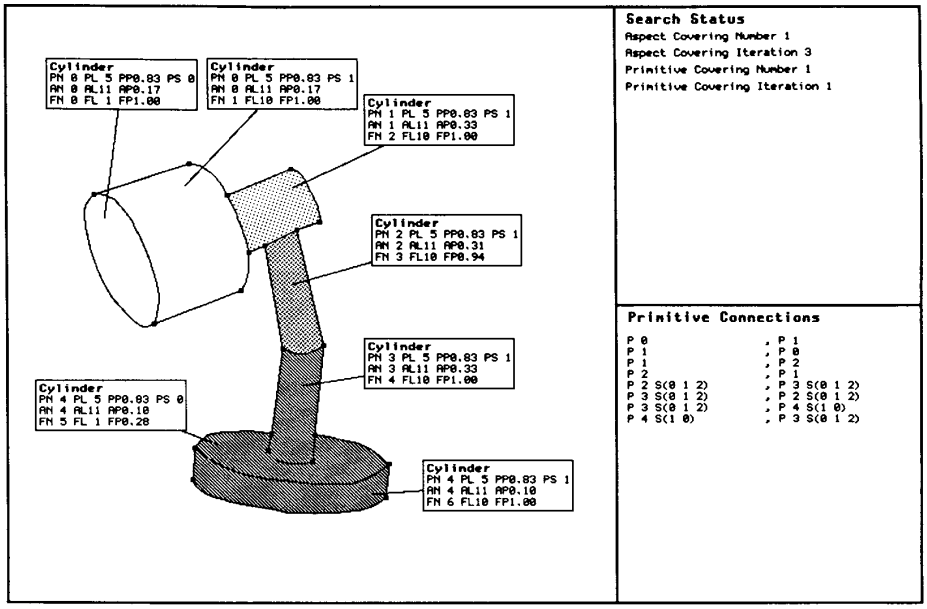

Fig. 22. Correct first interpretation of a desk lamp.

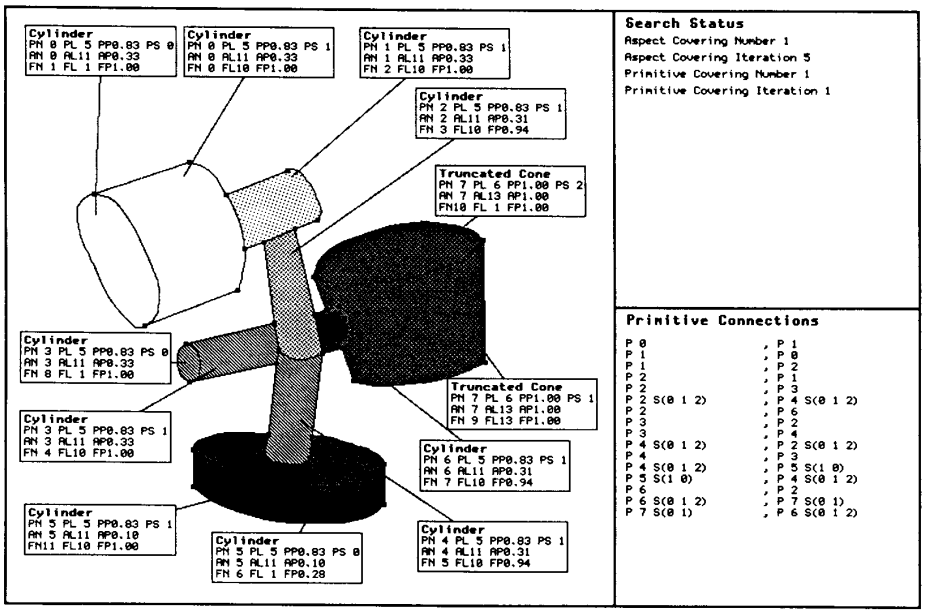

Fig. 23. Correct first interpretation of an occluded scene.

skeleton image. Despite the underpartitioning of the contours (contour 11), the first aspect and primitive coverings represent the correct interpretation of the scene.

In the second example, the entire process was applied to the 256 × 256 image of a padlock as shown in Fig. 26. Excluding the time required to transform the raw image into a skeleton image (140 s), the time required to generate the first primitive covering was 62 s. Fig. 27 shows the partitioned contours extracted from the skeleton image. Faces FN2, FN3, and FN4 were correctly grouped and interpreted as the block primitive (PL1). However, faces FN0 and FN1 were grouped and interpreted as the truncated ellipsoid primitive (PL8). In face FN1, the contours have been overpartitioned; therefore, the face could not be matched to the faces in the aspect hierarchy. Furthermore, due to noise, contour 2 was classified as a line. The strongest face inference given the possible boundary groups was the face labeled 15 (FL15) inferred from the boundary group (labeled 9 in Fig. 4) consisting of contours 0 and 4. When grouped with the elliptical face FN0, the resulting aspect was mapped to the truncated ellipsoid.

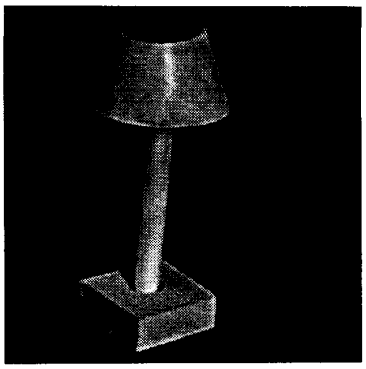

Fig. 24. Image of a table lamp (256 × 256).

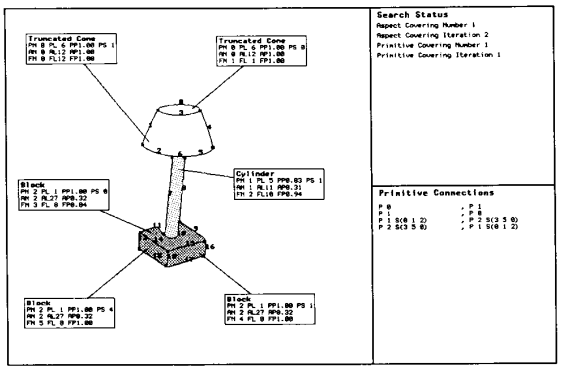

Fig. 25. Correct first interpretation of a table lamp.

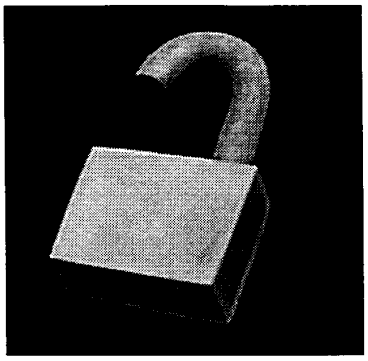

Fig. 26. Image of a lock (256 × 256).

The second strongest face inference (also from contours 0 and 4, boundary group label 9) was the face labeled 10 (FL10). The second aspect covering from which the correct primitive covering was inferred is shown in Fig. 28. The time taken to generate the second covering was an additional 2 s.

These examples illustrate the results of applying our shape recovery algorithm to real images. Despite the use of simple, standard techniques for image segmentation and contour grouping, which resulted in several segmentation errors, the algorithm was able to produce a correct interpretation of these scenes. With more effective techniques for region extraction (e.g., [34], [7]), contour partitioning (e.g., [49], [21]), perceptual grouping (e.g., [35], [33]), and the model-based segmentation correction of Section IV, we expect the system's performance to improve significantly. In addition, with a more efficient implementation on a faster target machine (such as a Sun™ Sparcstation 2 with a standard image preprocessor for operations such as filtering, line/region finding, etc.), we expect up to two orders of magnitude speedup.

VI. Related Work

In this section, we briefly review a number of related approaches and compare them with our own approach. The review is by no means complete; it is meant only to contrast our approach with some of the more established techniques. More comprehensive reviews can be found in [10], [6] and [14].

Many researchers use volumetric primitives to model objects. An often used class of primitives is the class of generalized cylinders [9], [1], [37], [11] whose cross section, axis, and sweep properties are arbitrary functions. Superquadrics [22] provide a volumetric representation requiring only a few parameters. Pentland [39] first applied superquadrics to primitive modeling for object recognition, whereas Pentland [40], Solina [51], and Terzopoulos and Metaxas [58] have achieved considerable success in deriving superquadric primitives from range data. Active or physically based models have been used by Terzopoulos *et al.* [56], [57] and Pentland [42]–[45] to successfully recover 3-D shape and nonrigid motion from natural imagery. Although generalized cylinders, superquadrics, and active models provide a rich language for describing parts, their extraction from the image is computationally complex.

Brooks' ACRONYM system [11] exemplifies the object-centered approach to object recognition. In ACRONYM, objects are represented as constructions of generalized cylinders. Recognition of a particular model object consists of predicting the projected appearance in the image of the object's components; constraints on the 3-D parts of the model are mapped to constraints on the 2-D parts of the projection. The image contours are then examined, subject to these constraints, and matched contours are used to further constrain the size and orientation of the 3-D parts. The top-down nature of ACRONYM makes it unsuitable for unexpected-object recognition; ACRONYM can only confirm or deny the existence in the image of a user-specified object. In addition, the quantitative nature of ACRONYM's constraints requires the overhead of a complex constraint manipulation system. ACRONYM is appropriate for recognizing subtypes of a particular type of aircraft, which our system cannot handle; however, in distinguishing an aircraft from, say, a horse, we avoid detailed quantitative constraints.

In contrast with ACRONYM's top-down approach, Lowe's SCERPO system [33] takes a more bottom-up approach to object-centered recognition. In SCERPO, objects are represented as polyhedra or constructions of 3-D faces. Image contours are first grouped according to perceptual organization rules, including parallelism, symmetry, and collinearity. From these groupings, simple 3-D inferences are made about the 3-D contours comprising the object; for example, parallel

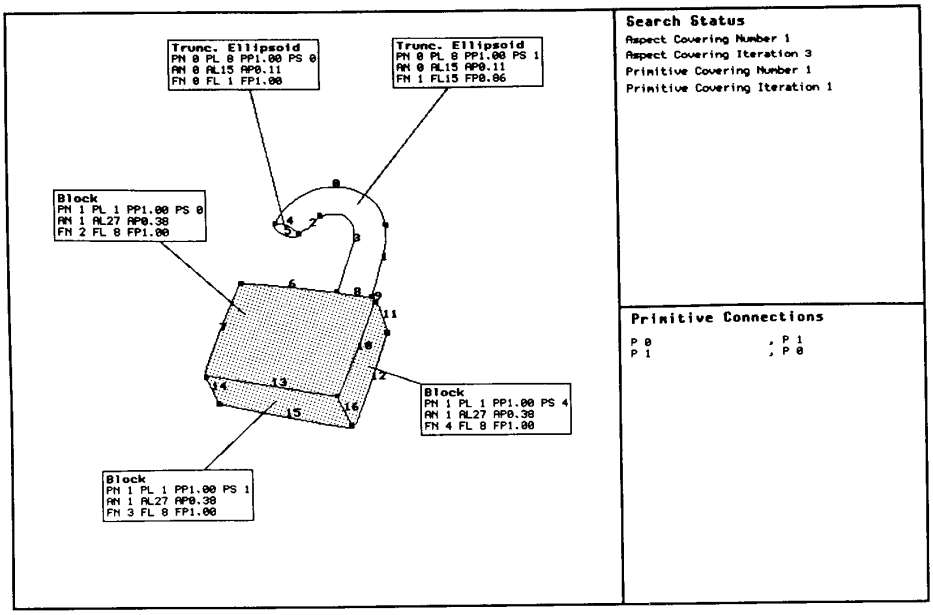

Fig. 27. Incorrect first interpretation of the lock.

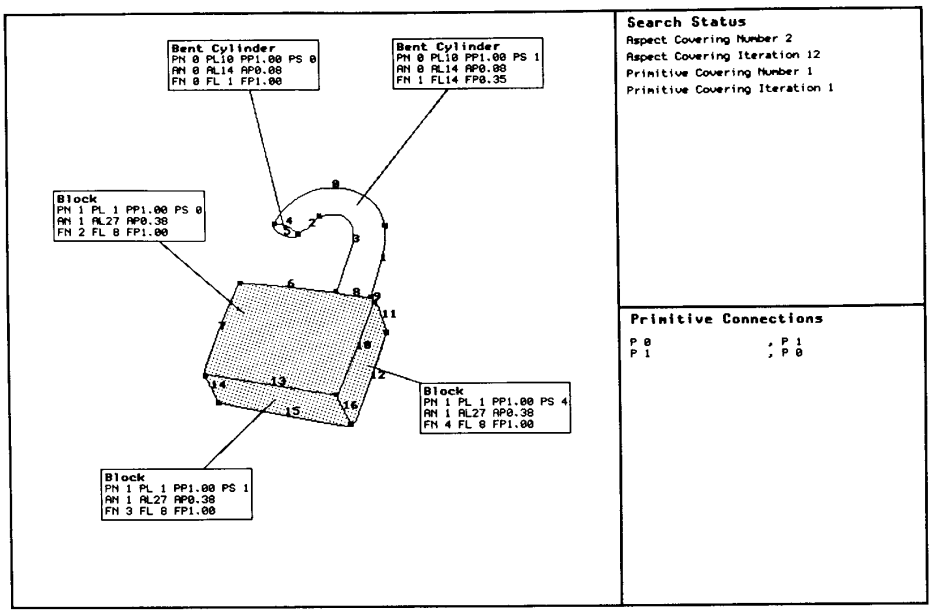

Fig. 28. Correct interpretation of the lock.

lines in the image imply parallel edges in the polyhedral object. The 3-D inferences are matched against manually identified instances of the properties in the model. Back-projected features are used to verify the object and constrain its position and orientation. Although SCERPO could be applied to unexpected-object recognition, the complexity of polyhedral models and the simplicity of the indexing features result in large indexing ambiguity. In addition, SCERPO's polyhedral models restrict its recognition domain to rigid objects. Our modeling scheme and indexing primitives, on the other hand, support the recognition of articulated objects.

Modeling objects using a set of qualitative primitives is not new. Mulgaonkar *et al.* [36] describe a recognition system based on a set of generalized blob models including sticks, plates, and blobs. From the 2-D silhouette of an object, a graph-theoretic clustering technique yields a set of convex polygonal parts; internal image contours are ignored. The projected parts are then compared with 3-D part instances in the model database and are subject to quantitative geometric and relational constraints. Like ACRONYM, the system is primarily top-down, starting with a model and matching image structures to the model-based predictions.

Biederman [8] proposed a set of primitives, called geons, based on dichotomous and trichotomous properties of generalized cylinders. Bergevin and Levine [2]–[4] have applied Biederman's geons to 3-D object recognition from 2-D images in a system called PARVO. Their approach to grouping lines consists of pairing segmentation points resulting from concave slope discontinuities lying on the silhouette boundary of the object. From this pairing, line groups are formed, and internal contours are later assigned to the line groups on a second pass. Given a set of cross-sectional and body faces representing a line group, a discrimination tree is traversed to determine the symbolic values of the geon attributes. The technique assumes that the segmentation points can be paired and assumes that a unique geon label can be assigned to each group of lines constituting a part. However, in the presence of occlusion or degenerate viewpoint, these assumptions may not be correct. The major disadvantage of their approach is that it is dependent on their choice of geons as modeling primitives.

The viewer-centered representation of an object by a set of aspects was applied to 3-D object recognition by Chakravarty and Freeman [13] and more recently by Ikeuchi and Kanade [28], Fan *et al.* [20], Stark *et al.* [53], and Shapiro and Lu [50]. These approaches model the entire object using a set of aspects. The number of aspects required to represent the objects in a database is proportional to the number of objects in the database and to object model complexity; in our case, the number of aspects is finite. Rather than matching image features to a large number of complex aspects, we identify local instances of simple aspects. This allows us, like ACRONYM, to have articulated models since we are matching aspects to primitives rather than to objects.

Another recent approach to 3-D object recognition using viewer-centered models has been proposed by Ullman and Basri [60]. In this approach, a 3-D object is represented by a linear combination of a small number of 2-D images of the object. A major advantage of this approach is the ease in which object models can be acquired. Unfortunately, each topologically distinct view (aspect) of the object requires a distinct set of images. For large databases containing complex objects, the number of views could become prohibitive.

VII. Discussion

The inefficiency of most 3-D object recognition systems is reflected in the relatively small number of objects in their databases (on the order of 10); in many cases, algorithms are demonstrated on a single object model. The major problem is that these systems terminate the bottom-up primitive extraction phase very early, resulting in simple primitives such as lines, corners, and inflections. These primitives do not provide very discriminating indices into a large database; therefore, there are normally a large number of hypothesized matches. Consequently, the burden of recognition falls on top-down verification, which, for simple geometric image features, requires both accurate estimates of the object's pose and prior knowledge of the object's geometry.

We instead index into the model database with more discriminating primitives (ones that do not require precise knowledge of model geometry or accurate estimates of pose). An appropriate choice for higher order indexing primitives is the class of volumetric primitives that capture the intuitive notion of an object's parts. In this approach, object models are constructed from object-centered 3-D volumetric primitives. The primitives, in turn, are represented in the image by a set of viewer-centered aspects.

Although any selection of volumetric primitives can be mapped to a set of aspects, our hierarchical aspect representation is particularly appropriate for primitives with distinct surfaces, i.e. primitives whose aspects contain distinct faces. The use of a face-based aspect hierarchy is the backbone of our approach, allowing us to obtain probabilistic rules for inferring more complex features from less complex features and for merging oversegmented contours and regions. Although the individual features represented in our aspect hierarchy may change when using other types of volumetric primitives, the concept of representing a set of 3-D volumetric primitives as a probabilistic hierarchy of image features is applicable to any object representation that models objects using volumetric parts.

The cost of extracting more complex primitives from the image is the difficulty of grouping less complex features into more complex features; the number of possible groupings is enormous. Our recovery algorithm uses a statistical analysis of the aspects to rank-order the possible groupings. The result is a heuristic that has been demonstrated to quickly arrive at the correct interpretation. Note, however, that our approach will, if need be, enumerate all possible interpretations (or groupings); the correct interpretation of any scene, no matter how ambiguous or unlikely, will eventually be generated.

Our recovery algorithm has been demonstrated to effectively recover a set of occluded 3-D volumetric primitives from an input scene, providing a powerful front end to a recognition system. In fact, we have developed a prototype of such a recognition system that has successfully recognized all the objects recovered in Section V [17]. A recognition system that can recognize objects based on their coarse structure could also serve as a front end to a more quantitative recognition system.

APPENDIX A
The Complexity of the Aspect Covering Problem

The *partition into isomorphic subgraphs* problem [23] is stated as follows: Given two graphs $G = (V, E)$ and $H = (V', E')$, can the vertices of G be partitioned into q disjoint sets $V_1, V_2, V_3, \ldots, V_q$ such that, for $1 \leq i \leq q$, the subgraph of G induced by V_i is isomorphic to H? Kirpatrick and Hell [29] prove that this problem remains n-p complete for any fixed H, $|H| \geq 3$. Whether or not the problem is n-p complete for planar G and H remains an open problem. However, Berman *et al.* [5] have recently shown that the problem is n-p complete for any connected outerplanar H, $|H| \geq 4$ and that the problem is solvable in linear time for any triangulated H, $|H| \geq 4$. Since our aspect face graphs are neither outerplanar nor triangulated, the complexity of our problem remains open. In any case, we know of no polynomial time solution to this problem.

If we assume that the *partition into isomorphic subgraphs* problem (for fixed H, planar G, and H) is n-p complete, we can prove that our problem is n-p complete. We begin by reducing an instance of *partition into isomorphic subgraphs* into an instance of the decision problem: Does there exist an aspect covering of the face graph? The reduction is simple. Let G become a face graph FG in our problem, and let H become A_1, which is the graph representing the singleton aspect defined in an aspect hierarchy; both of these mappings are polynomial in the number of nodes in G. Thus, if we could solve our decision problem in polynomial time, we could solve the "partition into isomorphic subgraphs" problem (for fixed H, planar G, and H) in polynomial time, making our problem n-p hard. Since our problem is in n-p (generate both a partition and a labeling, and check their consistency in polynomial time), our decision problem is n-p complete. Since our actual problem of finding an aspect covering is at least as hard as the corresponding decision problem and is also in n-p, the problem of finding an aspect covering is n-p complete.

APPENDIX B
ASPECT INSTANTIATION

Consider a face graph FG and an aspect hypothesis ah with label t seeded at face f in FG. The aspect hierarchy aspect corresponding to label t, herein called the aspect definition, specifies that the aspect contains k faces, where each has a specified label and adjacency relations. We first collect together all neighboring faces of f (including f itself) in FG. Next, we generate all face subsets of size $\leq k$ from this collection; recall that there is an upper bound on k that is fixed (specified by the aspect hierarchy) and independent of the size of FG. For each subset, we check to see if the subgraph of FG (i.e., face subgraph) induced by the face subset is isomorphic to the aspect definition. For each matching subset, we instantiate an aspect; the result is a (possibly empty) list of instantiated aspects.

An aspect can be instantiated from an aspect hypothesis and a face subgraph if and only if the following conditions are satisfied:

1) For each face in the face subgraph, there must exist, among its list of face hypotheses, a hypothesis whose label agrees with the label of its matching face in the aspect definition; if such a face hypothesis is found, it is *assigned* to the face in the subgraph.
2) For each arc (or face adjacency relation) in the face subgraph, there must exist a corresponding arc in the aspect definition. Similarly, for each arc in the aspect definition, there must exist a corresponding arc in the face subgraph.
3) For each arc in the face subgraph involving two faces A and B, there must exist a seed contour set belonging to the face hypothesis assigned to A and a seed contour set belonging to the face hypothesis assigned to B such that each of the two seed contour sets includes the contours shared by A and B. More intuitively, the contour(s) shared by two faces must be seed contours of both faces.
4) For each face in the face subgraph, there must exist at least one seed contour set belonging to its assigned face hypothesis that satisfies all face adjacency relations involving that face.

If an aspect with k faces cannot be instantiated from an aspect hypothesis, it may be due to the fact that the aspect is occluded in the image. In this case, our goal is to find subsets of image faces that match portions of the aspect definition. Consider the set S of all subsets of image faces such that for each s in S, $|s| < k$, and the face subgraph induced by s matches some portion of the aspect definition (according to the above set of conditions). In addition, according to the partial match, let the valid seed contour sets at face i in s be $SC_1^i, SC_2^i, \ldots, SC_w^i$. Finally, let r represent the faces in the aspect definition not included in s (presumably occluded). We instantiate the aspect encompassing s, provided the following conditions are satisfied:

1) There exists no other subset t in S such that s is a proper subset of t and the aspect encompassing t has been instantiated. More intuitively, if a set of faces satisfies an aspect, we ignore its subsets (which may also satisfy the aspect).
2) For each arc in the face graph involving a face fs in s and a face fr in r, there must exist a valid seed contour set SC_j^{fs} belonging to the face hypothesis assigned to fs such that the contours shared by fs and fr do not appear in the seed contour set. More intuitively, if face A is occluded by face B, then the contours shared by faces A and B (which belong to face B) should not be seed contours of face A.

The above restrictions have a significant impact on the selection of boundary groups. If we have a weak (i.e., low probability) face hypothesis, then it is likely that each of its seed contour sets represents a small fraction of the contours comprising the face. Consequently, instantiation of an aspect including such a face hypothesis may fail since it is likely that the required neighboring faces do not border at seed contours. However, with the lack of seed contours, smaller subgraphs may match the aspect definition since it is likely that neighboring faces do not border at seed contours. We conclude that there is a tradeoff between selecting only the best boundary groups and exhaustively selecting all boundary groups. In the former case, a strong face hypothesis supported by strong boundary groups will likely match few aspect definitions, pruning out many interpretations of the face. However, if weaker boundary groups are not included in the face hypothesis, a correct interpretation may be impossible. Conversely, the presence of weak boundary groups allows occluded aspects to be instantiated. Although this may guarantee a solution, the increased number of interpretations may lengthen the search for a solution and may result in less likely solutions being prematurely generated.

ACKNOWLEDGMENT

The authors would like to thank L. Davis, P. Cucka, N. Leibovic, and especially S. Stevenson for insightful discussions and for their comments on earlier drafts of this paper.

REFERENCES

[1] G. J. Agin and T. O. Binford, "Computer description of curved objects," *IEEE Trans. Comput.*, vol. C-25, no. 4, pp. 439–449, Apr. 1976.

[2] R. Bergevin and M. D. Levine, "Recognition of 3-D objects in 2-D line drawings: An approach based on geons," Tech. Rep. TR-CIM-88–24, McGill Res. Cent. Intell. Machines, McGill Univ., Nov. 1988.

[3] ——, "Hierarchical decomposition of objects in line drawings," Tech. Rep. TR-CIM-88–25, McGill Res. Cent. Intell. Machines, McGill Univ., Nov. 1988.

[4] ——, "Generic object recognition: Building coarse 3D descriptions from line drawings," in *Proc. IEEE Workshop Interpretation 3D Scenes* (Austin, TX), Nov. 1989, pp. 68–74.

[5] F. Berman, D. Johnson, T. Leighton, P. W. Shor and L. Snyder, "Generalized planar matching," *J. Algorithms*, vol. 11, no. 2, pp. 153–184, June 1990.

[6] P. J. Besl and R. C. Jain, "Three-dimensional object recognition," *ACM Comput. Surveys*, vol. 17, no. 1, pp. 75–145, Mar. 1985.

[7] ——, "Segmentation through variable-order surface fitting," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, no. 2, pp. 167–192, Mar. 1988.

[8] I. Biederman, "Human image understanding: Recent research and a theory," *Comput. Vision Graphics Image Processing*, vol. 32, pp. 29–73, 1985.

[9] T. O. Binford, "Visual perception by computer," in *Proc. IEEE Conf. Syst. Contr.* (Miami, FL), 1971.

[10] ——, "Survey of model-based image analysis systems," Int. J. Robotics Res., vol. 1, no. 1, pp. 18–64, Spring 1982.

[11] R. A. Brooks, "Model-based 3-D interpretations of 2-D images," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 5, no. 2, pp. 140–150, Mar. 1983.

[12] J. Canny, "A computational approach to edge detection," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 8, no. 6, pp. 679–698, Nov. 1986.

[13] I. Chakravarty and H. Freeman, "Characteristic views as a basis for three-dimensional object recognition," in *Proc. SPIE Conf. Robot Vision* (Arlington, VA), May 6–7, 1982, pp. 37–45.

[14] R. T. Chin and C. R. Dyer, "Model-based recognition in robot vision," *ACM Comput. Surveys*, vol. 18, no. 1, pp. 67–108, Mar. 1986.

[15] S. J. Dickinson, A. P. Pentland, and A. Rosenfeld, "A representation for qualitative 3-D object recognition integrating object-centered and viewer-centered models," in *Vision: A Convergence of Disciplines* (K.N. Leibovic, Ed.). New York: Springer Verlag, 1990.

[16] ——, "Qualitative 3-D shape recovery using distributed aspect matching," Tech. Rep. CAR-TR-453, Cent. Automat. Res., Univ. Maryland, June 1990.

[17] ——, "From volumes to views: An approach to 3-D object recognition," in *Proc. IEEE Workshop CAD-Based Vision* (Maui, HI), June 1991.

[18] D. Eggert and K. Bowyer, "Computing the orthographic projection aspect graph of solids of revolution," in *Proc. IEEE Workshop Interpretation 3D Scenes* (Austin, TX), 1989, pp. 102–108.

[19] T. J. Fan, G. Medioni, and R. Nevatia, "Recognizing 3-D objects using surface descriptions," in *Proc. Second Int. Conf. Comput. Vision* (Tampa, FL), 1988, pp. 474–481.

[20] M. A. Fischler and R. C. Bolles, "Perceptual organization and curve partitioning," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 8, no. 1, pp. 100–105, Jan. 1986.

[21] M. Gardiner, "The superellipse: A curve that lies between the ellipse and the rectangle," *Sci. Amer.*, vol. 213, pp. 222–234, Sept. 1965.

[22] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman, 1979, p. 193.

[23] Z. Gigus and J. Malik, "Computing the aspect graph for line drawings of polyhedral objects," Tech. Rep. UCB/CSD 88/402, Comput. Sci. Div. (EECS), Univ. of California, Berkeley, Feb. 1988.

[24] Z. Gigus, J. Canny, and R. Seidel, "Efficiently computing and representing aspect graphs of polyhedral objects," Tech. Rep. UCB/CSD 88/432, Comput. Sci. Div. (EECS), Univ. of California, Berkeley, Aug. 1988.

[25] R. M. Haralick and L. Shapiro, "Image segmentation techniques," *Comput. Vision Graphics Image Processing*, vol. 29, pp. 100–132, 1985.

[26] D. P. Huttenlocher and S. Ullman, "Object recognition using alignment," in *Proc. First Int. Conf. Comput. Vision* (London, UK), 1987, pp. 102–111.

[27] K. Ikeuchi and T. Kanade, "Automatic generation of object recognition programs," *Proc. IEEE*, vol. 76, no. 8, pp. 1016–1035, Aug. 1988.

[28] D. G. Kirpatrick and P. Hell, "On the complexity of a generalized matching problem," in *Proc. 10th Ann. ACM Symp. Theory Comput.*, 1978, pp. 240–245.

[29] J. J. Koenderink and A. J. van Doorn, "The internal representation of solid shape with respect to vision," *Biol. Cybern.*, vol. 32, pp. 211–216, 1979.

[30] Y. Lamdan, J. T. Schwartz and H. J. Wolfson, "On recognition of 3-D objects from 2-D images," in *Proc. IEEE Int. Conf. Robotics Automat.* (Philadelphia, PA), 1988, pp. 1407–1413.

[31] Y. Liao, "A two-stage method of fitting conic arcs and straight-line segments to digitized contours," in *Proc. Patt. Recogn. Image Processing Conf.* (Dallas, TX), 1981, pp. 224–229.

[32] D. G. Lowe, *Perceptual Organization and Visual Recognition*. Norwell, MA: Kluwer, 1985.

[33] P. Meer, D. Mintz, and A. Rosenfeld, "Least median of squares based robust analysis of image structure," in *Proc. DARPA Image Understanding Workshop* (Pittsburgh, PA), 1990.

[34] R. Mohan and R. Nevatia, "Perceptual organization for segmentation and description," in *Proc. DARPA Image Understanding Workshop* (Palo Alto, CA), 1989, pp. 415–424.

[35] P. G. Mulgaonkar, L. G Shapiro, and R. M. Haralick, "Matching 'sticks, plates and blobs' objects using geometric and relational constraints," *Image Vision Comput.*, vol. 2, no. 2, pp. 85–98, May 1984.

[36] R. Nevatia and T. O. Binford, "Description and recognition of curved objects," *Artificial Intell.*, vol. 8, pp. 77–98, 1977.

[37] N. J. Nilsson, *Principles of Artificial Intelligence*. Los Altos, CA: Morgan Kaufmann, 1980, ch. 2.

[38] A. P. Pentland, "Perceptual organization and the representation of natural form," *Artificial Intell.*, vol. 28, pp. 293–331, 1986.

[39] ——, "Recognition by parts," in *Proc. First Int. Conf. Comput. Vision* (London, UK), 1987, pp. 612–620.

[40] ——, "Toward an ideal 3-D CAD system," in *Proc. SPIE Conf. Machine Vision Man-Machine Interface* (San Diego, CA), 1987.

[41] A. Pentland and J. Williams, "Good vibrations: Modal dynamics for graphics and animation," *ACM Comput. Graphics*, vol. 23, no. 4, pp. 215–222, 1989.

[42] A. Pentland, "Automatic extraction of deformable part models," *Int. J. Comput. Vision*, vol. 4, pp. 107–126, 1990.

[43] A. Pentland and S. Sclaroff, "Closed-form solutions for physically based shape modeling and recognition," to be published in *IEEE Trans. Patt. Anal. Machine Intell.*.

[44] A. Pentland and B. Horowitz, "Recovery of nonrigid motion and structure," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 7, pp. 730–742, July 1991.

[45] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Comput. Graphics Image Processing*, vol. 1, pp. 244–256, 1972.

[46] L. G. Roberts, "Machine perception of three-dimensional solids," in *Optical and Electro-Optical Information Processing*(J.T. Tippett *et al.*, Eds.). Cambridge, MA: MIT Press, 1965, pp. 159–197.

[47] A. Rosenfeld, "Recognizing unexpected objects: A proposed approach," in *Proc. DARPA Image Understanding Workshop* (Los Angeles, CA), 1987, pp. 620–627.

[48] P. Saint-Marc and G. Medioni, "Adaptive smoothing for feature extraction," in *Proc. 1988 DARPA Image Understanding Workshop* (Cambridge, MA), 1988, pp. 1100–1113.

[49] L. G. Shapiro and H. Lu, "Accumulator-based inexact matching using relational summaries," *Machine Vision and Applications*, vol. 3, pp. 143–158, 1990.

[50] F. Solina, "Shape recovery and segmentation with deformable part models," Tech. Rep. MS-CIS-87–111, Univ. of Pennsylvania, Philadelphia, PA, Dec. 1987.

[51] T. Sripradisvarakul and R. Jain, "Generating aspect graphs for curved objects," in *Proc. IEEE Workshop Interpretation 3D Scenes* (Austin, TX), 1989, pp. 109–115.

[52] L. Stark, D. Eggert, and K. Bowyer, "Aspect graphs and nonlinear optimization in 3-D object recognition," in *Proc. IEEE Second Int. Conf. Comput. Vision* (Tampa, FL), 1988, pp. 501–507.

[53] J. Stewman and K. Bowyer, "Creating the perspective projection aspect graph of polyhedral objects," in *Proc. IEEE Second Int. Conf. Comput. Vision* (Tampa, FL), 1988, pp. 494–500.

[54] , "Direct construction of the perspective projection aspect graph of convex polyhedra," *Comput. Vision Graphics Image Processing*, vol. 51, pp. 20–37, 1990.

[55] D. Terzopoulos, A. Witkin, and M. Kass, "Symmetry-seeking models and 3D object recovery," *Int. J. Comput. Vision*, vol. 1, pp. 211–221, 1987.

[56] , "Constraints on deformable models: Recovering 3D shape and nonrigid motion," *Artificial Intell.*, vol. 36, pp. 91–123, 1988.

[57] D. Terzopoulos and D. Metaxas, "Dynamic 3D models with local and global deformations: Deformable superquadrics," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 7, pp. 703–714, July 1991.

[58] D. W. Thompson and J. L. Mundy, "Model-directed object recognition on the connection machine," in *Proc. DARPA Image Understanding*

Workshop (Los Angeles, CA), 1987, pp. 93–106.
[59] S. Ullman and R. Basri, "Recognition by linear combinations of models," A.I. Memo 1152, Artificial Intell. Lab., Mass. Inst. Technol., Aug. 1989.
[60] A. P. Witkin and J. M. Tenenbaum, "On the role of structure in vision," in *Human and Machine Vision* (J. Beck *et al.*, Eds.). New York: Academic, 1983, pp. 481–543.

Sven J. Dickinson received the B.A.Sc. degree in systems design engineering from the University of Waterloo, Waterloo, Canada, in 1983 and the M.S. and Ph.D. degrees in computer science from the Department of Computer Science, University of Maryland, in 1988 and 1991, repectively.

From 1985 to 1991, he was a Graduate Research Assistant in the Computer Vision Laboratory at the Center for Automation Research, University of Maryland. Prior to this, he worked in the computer vision industry, designing image processing systems for Grinnell Systems Inc., San Jose, CA, and optical character recognition systems for DEST, Inc., Milpitas, CA. He is currently a Research Associate at the Artificial Intelligence Laboratory, University of Toronto. His major field of interest is computer vision with an emphasis on object recognition.

Alex P. Pentland received the Ph.D. degree from the Massachusettes Institute of Technology (M. I. T.) in 1982 and began work at SRI's International Artificial Intelligence Center.

He was appointed Industrial Lecturer at Stanford University's Computer Science Department in 1983, winning the Distinguished Lecturer Award in 1986. In 1987, he was appointed Associate Professor of Computer, Information, and Design Technology at M. I. T's Media Laboratory and was appointed Associate Professor in the M. I. T. Civil Engineering Department in 1988. In addition, in 1988, he was appointed to the NEC Computer and Communications Career Development Chair. He has published more than 130 scientific articles in the fields of artificial intelligence, machine vision, design, and computer graphics.

In 1984, Dr. Pentland won the Best Paper Prize from the American Association for Artificial Intelligence for his research on problems of texture and shape description. In 1991, he won the Best Prize Paper from the IEEE for his work in face recognition. His last book was entitled *From Pixels to Predicates* (Norwood, NJ: Ablex), and he is curently working on a new book entitled *Dynamic Models for Vision* (Cambridge, MA: Bradford/M. I. T. Press).

Azriel Rosenfeld (F'72) received rabbinic ordination in 1952, the Doctor of Hebrew Literature degree from Yeshiva University, New York, NY, in 1955, the Ph.D degree in mathematics from Columbia University, New York, NY, in 1957, and an Honorary Doctor of Technology degree from Linkopking University, Sweden, in 1980.

He is currently a tenured Research Professor and Director of the Center for Automation Research, which is a department-level unit of the University of Maryland, College Park. He also holds affiliate professorships in the Departments of Computer Science and Psychology and in the College of Engineering. He is a widely known reseacher in the field of computer image analysis. He wrote the first textbook in the field in 1969, was a founding editor of the first journal in the field in 1972, and was co-chairman of the first international conference in the field in 1987. He has published more than 20 books and more than 400 book chapters and journal articles and has directed more than 40 Ph.D dissertations.

Dr. Rosenfeld won the IEEE Emanuel Piore award in 1985. He was a founding director of the Machine Vision Association of the Society of Manufacturing Engineers from 1985–1988 and won its President's Award in 1987. He was a founding member of the IEEE Computer Society's Technical Committee on Pattern Analysis and Machine Intelligence in 1965, served as its chairman from 1985–1987, and received the Society's Meritorious Service Award in 1986. He was a founding member of the Governing Board of the International Association for Pattern Recognition from 1978–1985, served as its president from 1980–1982, and won its first K. S. Fu Award in 1988. He is a fellow of the Washington Academy of Sciences (elected in 1988) and won its Mathematics and Computer Science Award in 1988. He is a Founding Fellow of the American Association for Artificial Intelligence (elected in 1990) and is a Corresponding Member of the National Academy of Engineering of Mexico (elected in 1982). He is also a Foreign Member of the Academy of Science of the German Democratic Republic (elected in 1988). He became a certified Manufacturing Engineer in 1988.