From Volumes to Views: An Approach to 3-D Object Recognition

Sven J. Dickinson[1] Alex P. Pentland[2]
Azriel Rosenfeld[1]

[1]Center for Automation Research, University of Maryland, College Park, MD 20742-3411
[2]Vision and Modeling Group, Media Laboratory, M.I.T., Cambridge, MA 02139

Abstract

We present an approach to the recovery and recognition of 3-D objects from a single 2-D image. Given a recognition domain consisting of a database of objects, we select a set of object-centered 3-D volumetric modeling primitives that can be used to construct the objects. Next, we take the set of primitives and generate a hierarchical aspect representation based on their projected surfaces; conditional probabilities capture the ambiguity of mappings between levels of the hierarchy. From a region segmentation of the input image, we present a novel formulation of the recovery problem based on grouping the regions into aspects. Once the aspects are recovered, we use the aspect hierarchy to infer a set of volumetric primitives and their connectivity. The recovered primitives are are then used as indices into the object database for recognition.

1 Introduction

Significant progress has been made in the feature-based recognition of three-dimensional objects from two-dimensional images; some important examples include Lowe [13], Huttenlocher and Ullman [8], Thompson and Mundy [19], and Lamdan et al. [12]. 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 [5]. 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 pro-

cess. 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.

From an interpretation of the scene, the final step involves recognizing the (possibly occluded) objects. The recovered primitives are in the form of a graph whose nodes represent the 3-D primitives and whose arcs represent possible connections between the primitives. From this primitive graph, subgraphs are used as indices into a precomputed hash table. Candidate objects are then verified and ranked according to a goodness of fit measure. In this paper, we describe the approach and demonstrate its application to both synthetic and real images.

2 Building the Search Tables

2.1 Choosing the 3-D Primitives

Given a database of object models representing the domain of a recognition task, we seek a set of three-dimensional 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 [13], generalized cylinders [3], and superquadrics [14]. 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 which 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.

To demonstrate our approach to primitive recovery, we have selected an object representation similar to that used by Biederman [2], in which the Cartesian product of contrastive shape properties gives rise to a set of volumetric primitives called geons. For our investigation, we have chosen three properties including cross-section shape, axis shape, and cross-section size variation. The values of these properties give rise to a set of ten qualitative volumetric primitives. To construct objects, the primitives are attached to one another with the restriction that any junction of two primitives involves exactly one attachment surface from each primitive.

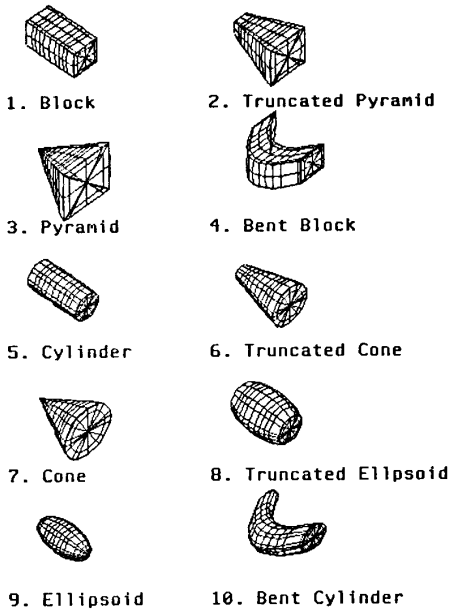

Figure 1: The Ten Primitives

In our system, these ten primitives were modeled using Pentland's SuperSketch 3-D modeling tool [14], as illustrated in Figure 1. 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. This is in contrast to Bergevin and Levine's PARVO system [1], whose approach is dependent on geons as a basis for object construction. Our approach can easily accommodate other sets of volumetric primitives.

2.2 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 [4, 6, 9, 10, 16, 17]. 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*. The aspect hierarchy consists of three levels, based on the faces appearing in the aspect set; Figure 2 illustrates a portion of the aspect hierarchy.

Aspects constitute the top level of the aspect hierarchy and consist of connected sets of faces. For our set of ten modeling primitives, there are 37 unique aspects [5]. Each 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.

Faces that make up the various aspects form the second level of the aspect hierarchy. For our set of 37 aspects, there are 16 unique faces. Each face 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. Reasoning about the type and arrangement of visible faces can allow identification of an aspect even when it is partially occluded.

Boundary Groups are subsets of the faces' bounding contours and make up the third and lowest level of the aspect hierarchy. For our set of 16 faces, there are 31 unique boundary groups. Each boundary group is represented by a graph in which nodes represent bounding contours and arcs represent nonaccidental contour relations. The boundary groups represent *qualitative* relationships among *qualitatively-described* contours; exact lengths, distances, angles, curvature, etc., are not represented. When a face is partially occluded, its boundary groups provide a mechanism for identifying the face type.

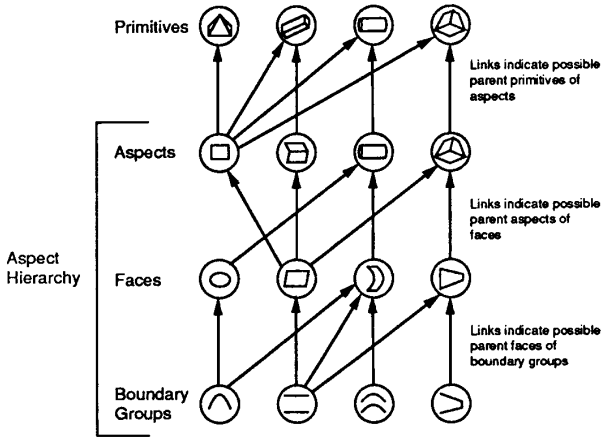

Figure 2: The Aspect Hierarchy

2.3 2-D Aspects to 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.

To generate these conditional probabilities, we first model our 3-D volumetric primitives using the SuperSketch modeling tool [14]. 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 result-

ing frequency distribution gives rise to the three conditional probability matrices (which can be found in [5]). 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 factored into the analysis. 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, e.g., [7, 11, 18].

3 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 the approach in greater detail.

3.1 Face Extraction

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. Once a set of contours has been extracted, 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. 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. The algorithm for transforming a contour graph into a face graph can be found in [5].

3.2 Face Labeling

Once the faces have been extracted, we must classify each face according to the faces in the aspect hierarchy. Recall that a face is represented by a graph in which nodes represent the face's contours and arcs represent relations between contours.[1] Thus, the classification of a face in the image consists of comparing its graph to those graphs representing the faces 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.

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.[2] 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.

3.3 Extracting Aspects

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 an *aspect 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.

There is no known polynomial time algorithm to solve this problem (see [5] for a discussion on the problem's complexity); however, the conditional 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

[1] Two adjacent collinear or curvilinear contours bounding a face may have been separated in the contour graph by a junction. 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.

[2] 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.

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 a detailed discussion of aspect instantiation and how occluded aspects can be instantiated, see [5].

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 which 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. For our search through the possible aspect labelings of the face graph, we employ Algorithm A with a heuristic based on the probabilities of the aspect hypotheses. Details of the algorithm can be found in [5].

3.4 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 which 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. 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 greater detail.

Algorithm for Enumerating Primitive Coverings. For each aspect in the aspect covering, we can use the aspect to primitive mapping to hypothesize a set of *primitives*. 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. To enumerate the selections, we employ a variation on the search algorithm used to enumerate the aspect coverings. The heuristic evaluation function negates the sum of the probabilities of the primitive, thereby favoring higher probability interpretations.

Extracting Primitive Connectivity. A primitive covering, represented by a graph in which nodes represent primitives and arcs represent primitive adjacencies, is then compared to 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).[3]

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.

A strong primitive connection strongly suggests the existence of a connection between two primitives. We

[3] 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.

can enhance the indexing power of a strongly connected subgraph if the attachment surfaces involved in each connection are hypothesized. Although it is impossible to define a set of domain independent rules which will, for any given set of primitives, correctly specify the attachment surfaces involved in a connection, we can define a set of heuristics which 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; details can be found in [5].

4 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, often this assumption will 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.

4.1 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. For if there exists a subgraph which is correctly labeled, i.e., a correctly labeled boundary group, the correct face hypothesis will be generated. Of course, the smaller the subgraph, the greater the number of different face hypotheses that will be generated, thereby increasing the space of labelings that must be examined during the aspect covering process.[4]

A more effective approach is to associate probabilities with the labeling of each contour (node) and contour relation (arc) in the graph [13]. 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.

Therefore our approach to the curve partitioning problem is to assume an *oversegmentation* of a face's bounding contours and use the aspect hierarchy to perform a model-based merging of adjacent contours during the face labeling stage. A contour oversegmentation can normally be obtained by using conservative parameters in most curve partitioning algorithms.

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 of c_f and c_s, provided that c_f and c_s are similar according to some criteria, and examine the resulting S.[5] 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.

4.2 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 the curve partitioning 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. A region oversegmentation can normally be obtained by using conservative parameters in most 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

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

[5] 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.

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.

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.

5 Object Recognition

Given a primitive graph representation of the scene, in which nodes represent 3-D volumetric primitives and arcs represent strong or weak connections between the primitives, the final task is to identify the object(s) in the scene. This task consists of two steps: 1) identifying possible candidate models that might be present in the scene (model indexing), and 2) verifying that these models appear in the scene. The following subsections discuss these steps in detail.

5.1 Model Indexing

The most inefficient model indexing strategy is to compare the primitive graph to each model in the object database; however, for large databases, the cost of verification may be prohibitive. Our goal has been to reconstruct from the image richer, more complex primitives whose combination offers a more discriminating index into the object database. Unlike simpler features such as lines, points, or corners which may be common to all objects, a collection of strongly connected primitives is unlikely to be common to many objects. Therefore, our model indexing strategy consists of first identifying all the strongly connected components in the primitive graph.[6]

The next logical step would be to construct a hash table in which each entry listed those models containing a particular strongly connected component. However, due to the variety of primitive labels and surface attachment specifications, and without a limit on the size of a strongly connected component, the size of such a hash table might be enormous. Consequently, we limit the size of the strongly connected components to three primitives.

For a given strongly connected component, the hash function returns a location in the hash table containing all object models (candidates) containing those primitive labels contained in the strongly connected component. The hash function currently ignores the connections between the primitives; these are examined during the model verification step. A more powerful hash function which does not limit the size of a strongly connected component and takes advantage of the primitive connections would result in a much larger hash table, appropriate for large object databases.

5.2 Model Verification

Given a set of candidate models corresponding to a given strongly connected component, the next step is to evaluate how well each model fits the scene (primitive covering). Since the hash function ignores the connections in the strongly connected component, the first step is to check that the strong connections in the strongly connected component exist between the corresponding primitives in each candidate model; this may result in some candidate models being discarded. At this point, for each remaining candidate model, the strongly connected component in the primitive covering is isomorphic to a subgraph of the candidate model. We then grow this correspondence according to the following steps:

1. Given a correspondence between a primitive covering subgraph PS and a model subgraph MS, we first choose a model primitive M_i that is not contained in the model subgraph, but is connected to a primitive M_j in the model subgraph. In PS, let the primitive corresponding to M_j be P_j.

[6] By strongly connected, we mean connected by *strong* arcs.

2. Among the neighbors (through strong or weak connections) of P_j in the primitive covering not contained in PS, select those whose label matches that of M_i. If more than one such neighbor exists, we create a new correspondence for each neighbor.

We repeat this sequence of steps for each correspondence until its size stabilizes. The entire process is then repeated for each strongly connected component in the primitive graph. The final result is a list of correspondences, each mapping a primitive covering subgraph to a model subgraph.

The final step ranks the correspondences according to a goodness of fit based on the probability of the verified primitives, the confidence of the primitive connections, and the attachment surfaces involved in the connections. Once the correspondences are ranked we choose the best correspondence and remove those aspects from the image that correspond to the recognized primitives. For the remaining aspects, we repeat the entire process. We first apply the primitive covering algorithm, extract strongly connected components, determine candidate models, and select the most likely candidate. The process is repeated until no aspects remain in the image. At any stage, a primitive covering may not yield any recognizable objects, i.e., candidate models. In this case, we generate a new primitive covering from the aspect covering and repeat the process; if all primitive coverings are exhausted, a new aspect covering is generated.

6 Results

We have built a system to demonstrate our approach to shape recovery and object recognition. The system has been implemented in LISP on a Symbolics^{TM} 3600. In the first example, we apply the approach to a manually segmented line drawing of a scene containing multiple occluded objects; all graphs representing image faces and boundary groups have been entered manually. The correct primitive covering is presented in the large box to the left of Figure 3; the faces have been shaded according to aspect. In this example, the first aspect and primitive coverings represent the correct interpretation of the scene. The time required to generate and recognize the recovered primitives was 40 seconds.

Each face in the image is described by a small box containing some mnemonics. The mnemonics PN, PL, PP, and PS refer to the primitive number (simply an enumeration of the primitives in the covering), primitive label (see Figure 1), primitive probability, and primitive attachment surface respectively, of the primitive that includes that face. The mnemonics AN, AL, and AP refer to the aspect number (an enumeration), aspect label, and aspect probability, respectively, of the aspect from which the primitive was inferred. The mnemonics FN, FL, and FP refer to the face number (an enumeration), face label, 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 addition, this box lists all objects currently (at These iterations) identified in the image, including their corresponding primitive numbers (PN) and goodness of fit; the object database contains 7 objects. Note that the pot handle was interpreted as a separate object; no collinearity grouping of primitives was performed. Therefore, the pot handle matched all objects in the database containing the cylinder primitive. The smaller box to the lower right indicates the primitive connections by primitive number (PN); if two primitives are strongly connected, a list of probable attachment surfaces appears in parentheses next to the primitive number. Note that this list is not exclusive, but rather a list of likely candidates.

For the next two examples, we apply the approach to real images containing objects. The first step in the process is the segmentation of the image into homogeneous regions. To extract regions, we first apply the Canny edge detector 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 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^{TM} 3 in approximately 140 seconds.

From the image of contours, we apply a connected components algorithm to extract a set of contours, each beginning and ending 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 [15] 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. 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

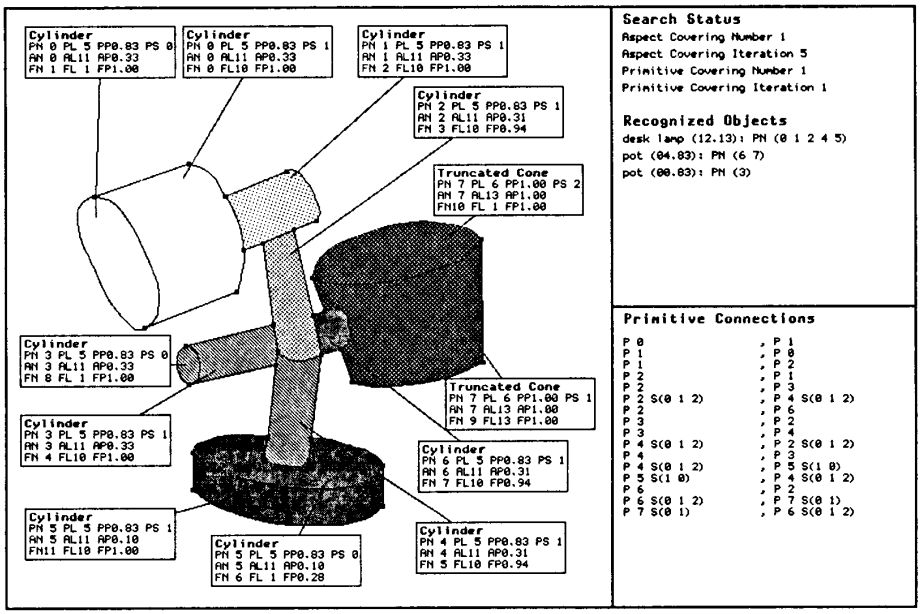

Figure 3: Correct First Interpretation of Occluded Scene

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 build the face graph using the algorithm described in [5]. 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 which meet at a junction are merged (according to the criteria used to check initial partition points) if they are collinear or curvilinear. Any curves bounding the face are further classified as concave or convex. Non-adjacent lines are labeled parallel if the angle between them is small, and symmetric if they are opposite and non-parallel. Non-adjacent 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 x 256 image of a table lamp as shown in Figure 6. Figure 5 represents the first primitive covering based on the first aspect covering. Excluding the time required to transform the raw image into a skeleton image, the time required to generate and recognize the first primitive covering was 62 seconds. 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 x 256 image of a padlock as shown in Figure 6. Excluding the time required to transform the raw image into a skeleton image, the time required to generate and recognize the correct primitive covering was 66 seconds. Figure 7 represents the first primitive covering based on the second aspect covering. However, the first primitive covering based on the first aspect covering was incorrect. In the first aspect covering, Faces FN2, FN3, and FN4 were correctly grouped and later 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

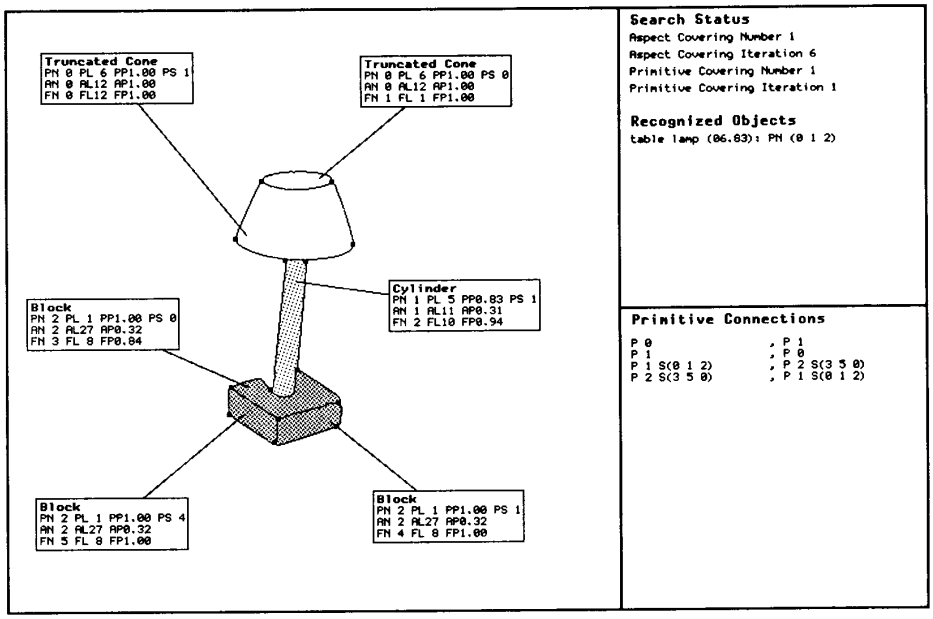

Figure 5: Correct First Interpretation of a Table Lamp

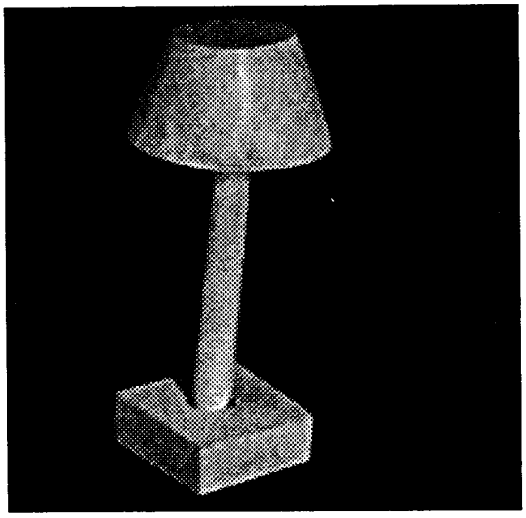

Figure 4: Image of a Table Lamp (256 x 256)

classified as a line. The strongest face inference, representing the projection of the body of a truncated ellipsoid, was generated by the boundary group consisting of contours 0 and 4. When grouped with the elliptical face FN0, the resulting aspect was mapped to the truncated ellipsoid. The second strongest (correct) face inference, representing the projection of the body of a bent cylinder, was also generated by contours 0 and 4. Therefore, the correct aspect label for FN1 was not the most probable and consequently did not take part in the first aspect covering.

These examples illustrate the results of applying our shape recovery and recognition 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, contour partitioning, perceptual grouping, and the model-based segmentation scheme of Section 4, 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^{TM} Sparcstation 2 with a standard image preprocessor for operations such as filtering, line finding, etc.) we expect up to two orders of magnitude speedup.

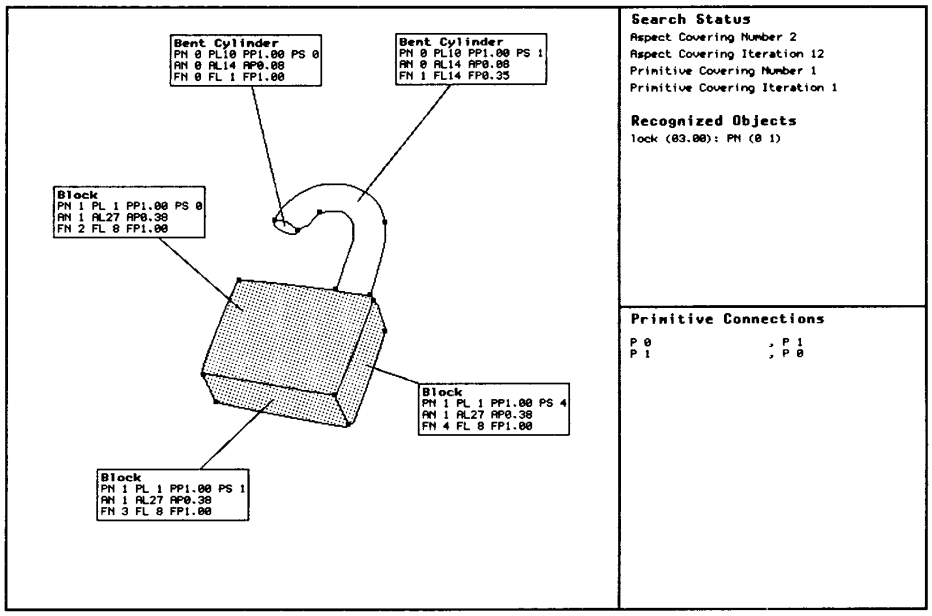

Figure 7: Correct Interpretation of the Lock

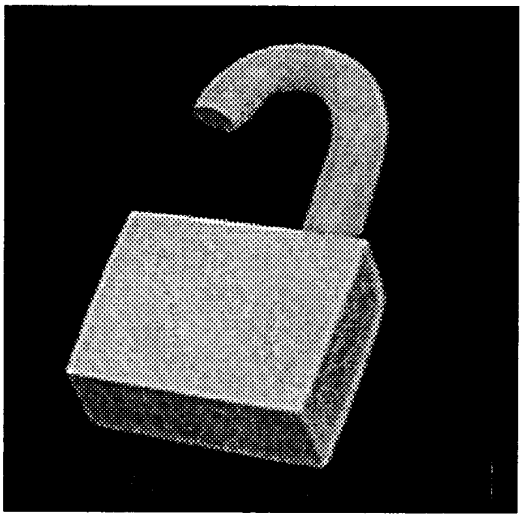

Figure 6: Image of a Lock (256 x 256)

7 Discussion

Previous recognition systems have terminated 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, so that there are a large number of hypothesized matches, each of which must be carefully verified.

Extracting more complex primitives, however, requires grouping less complex features, and 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.

Once a set of primitives has been recovered from the scene, visibly connected subsets of primitives offer powerful indices to an object database. Furthermore, unlike typical object recognition paradigms which verify geometrical image features, thereby requiring accurate pose determination, our approach performs a topologi-

cal verification of a graph representing the coarse structure of the object.

References

[1] Bergevin, Robert and Martin Levine, "Generic Object Recognition: Building Coarse 3D Descriptions from Line Drawings", Proceedings, IEEE Workshop on Interpretation of 3D Scenes, Austin, TX, November 1989, pp 68–74.

[2] Biederman, Irving, "Human Image Understanding: Recent Research and a Theory", *Computer Vision, Graphics, and Image Processing*, Vol. 32, 1985, pp 29–73.

[3] Brooks, Rodney Allen, "Model-Based 3-D Interpretations of 2-D Images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 5, No. 2, March 1983, pp 140–150.

[4] Chakravarty, Indranil and Herbert Freeman, "Characteristic Views as a Basis for Three-Dimensional Object Recognition", Proceedings, SPIE Conference on Robot Vision, Arlington VA, 1982, pp 37–45.

[5] Dickinson, Sven J., Alex P. Pentland, and Azriel Rosenfeld, "Qualitative 3-D Shape Recovery Using Distributed Aspect Matching", Technical Report CAR-TR-453, Center for Automation Research, University of Maryland, June, 1990.

[6] Fan, T.J., G. Medioni, and R. Nevatia, "Recognizing 3-D Objects using Surface Descriptions", Proceedings, IEEE Second International Conference on Computer Vision, Tampa, FL, 1988, pp 474–481.

[7] Gigus, Ziv, John Canny, and Raimund Seidel, "Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects", Proceedings, IEEE Second International Conference on Computer Vision, Tampa, FL, 1988, pp 30–39.

[8] Huttenlocher, Daniel P. and Shimon Ullman, "Object Recognition Using Alignment", Proceedings, First International Conference on Computer Vision, London, UK, 1987, pp 102–111.

[9] Ikeuchi, Katsushi and Takeo Kanade, "Automatic Generation of Object Recognition Programs", *Proceedings of the IEEE*, Vol. 76, No. 8, August 1988, pp 1016–1035.

[10] Koenderink, J. J. and A. J. van Doorn, "The Internal Representation of Solid Shape with Respect to Vision", *Biological Cybernetics*, Vol. 32, 1979, pp 211–216.

[11] Kriegman, David J. and Jean Ponce, "Computing Exact Aspect Graphs of Curved Objects: Solids of Revolution" *International Journal of Computer Vision*, Vol. 5, No. 2, 1990, pp 119–135.

[12] Lamdan, Yehezkel, Jacob T. Schwartz and Haim J. Wolfson, "On Recognition of 3-D Objects from 2-D Images", Proceedings, IEEE International Conference on Robotics and Automation, Philadelphia, PA, 1988, pp 1407–1413.

[13] Lowe, David G., *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Norwell, MA, 1985.

[14] Pentland, Alex P., "Perceptual Organization and the Representation of Natural Form", *Artificial Intelligence*, Vol. 28, 1986, pp 293–331.

[15] Ramer, U., "An Iterative Procedure for the Polygonal Approximation of Plane Curves", *Computer Graphics and Image Processing*, Vol. 1, 1972, pp 244–256.

[16] Shapiro, Linda G. and Haiyuan Lu, "Accumulator-Based Inexact Matching using Relational Summaries", *Machine Vision and Applications*, Vol. 3, 1990, pp 143-158.

[17] Stark, Louise, David Eggert, and Kevin Bowyer, "Aspect Graphs and Nonlinear Optimization in 3-D Object Recognition", Proceedings, IEEE Second International Conference on Computer Vision, Tampa, FL, 1988, pp 501–507.

[18] Stewman, John and Kevin Bowyer, "Direct Construction of the Perspective Projection Aspect Graph of Convex Polyhedra", *Computer Vision, Graphics, and Image Processing*, Vol. 51, 1990, pp 20–37.

[19] Thompson, D.W. and J.L. Mundy, "Model-Directed Object Recognition on the Connection Machine", Proceedings, DARPA Image Understanding Workshop, Los Angeles, CA, 1987, pp 93–106.