13
A Representation for Qualitative 3-D Object Recognition Integrating Object-Centered and Viewer-Centered Models

SVEN J. DICKINSON, ALEX P. PENTLAND, AND
AZRIEL ROSENFELD

Introduction

In the context of computer vision, the recognition of three-dimensional objects typically consists of image capture, feature extraction, and object model matching. During the image capture phase, a camera senses the brightness at regularly spaced points, or pixels, in the image. The brightness at these points is quantized into discrete values; the two-dimensional array of quantized values forms a digital image, the input to the computer vision system. During the feature extraction phase, various algorithms are applied to the digital image to extract salient features such as lines, curves, or regions. The set of these features, represented by a data structure, is then compared to the database of object model data structures in an attempt to identify the object. Clearly, the type of features that need to be extracted from the image depends on the representation of objects in the database.

In most cases, the features extracted from the image are considerably less complex than the object models with which they are compared; as a result, many models may contain a particular type of feature. The comparison of sets of image features to object models actually consists of two phases. In the bottom-up, or indexing, phase each image feature serves as an index into the object model database to select candidate object models containing that feature. In the top-down, or verification, phase the candidate model(s) are used to verify feature interpretations and to constrain further feature extraction.

In three-dimensional (3-D) object recognition by computer, two important issues pertain to the representation of objects. The first issue is the choice between object-centered and viewer-centered representations. Object-centered representations model objects as constructions of 3-D primitives, such as planar faces or generalized cylinders. Recognition consists of matching image features to the predicted projections of specific 3-D model features, a process requiring the determination of the object's position and orientation with respect to the camera. Viewer-centered representations model objects as a set of 2-D characteristic views, or aspects. Recognition consists of matching image features against the set of aspects; the view most closely resembling the features in the image defines the object and its orientation. The major advantage of viewer-

centered recognition is that it reduces the 3-D recognition problem to a 2-D recognition problem. However, with each model object having potentially many aspects, matching becomes less efficient than with object-centered models.

The second issue in 3-D object recognition concerns the amount of detail inherent in object models. If a quantitative representation specifies the exact dimensions and shape of an object, then simple model-based verification procedures can be employed to confirm or reject object hypotheses. However, exact representations of objects result in complex object models. If a qualitative representation captures only the gross shape characteristics of an object, then model-based verification will fail to predict the exact location of the object's features. Instead, the bottom-up stage of recognition must be extended to extract higher order features with which to index into the database. The advantage of qualitative models is that they are less complex than quantitative models, and are invariant to minor changes in shape.

This chapter proposes a modeling paradigm for 3-D object recognition from two-dimensional (2-D) images integrating object-centered and viewer-centered models. Object models are constructions of 3-D volumetric primitives, offering an efficient indexing mechanism for large object databases. The 3-D primitives, in turn, are mapped into a set of viewer-centered aspects. During recognition, image contours are matched to the contours comprising an aspect, defining the primitive type and constraining its orientation. The 3-D primitive is then used to index into the object database. When objects are composed of multiple primitives, occluded primitives from a given viewpoint project to occluded, and hence incomplete, aspects in the image. To accommodate the matching of occluded aspects, we have developed a hierarchical aspect representation based on the visible faces of the primitives. At the top of the hierarchy lie connected face structures, while at the bottom lie contour features of the component faces. Thus, aspect inferences can be made from incomplete projections.

With a large database of objects, we might expect there to be a great variety of primitive shapes and sizes. Representing every single primitive with a different set of aspects would make matching image contours to primitive aspects intractable. To minimize the size of the aspect set, we constrain the aspects to be invariant to minor changes in primitive shape, forcing the primitives to be qualitative in nature. The size of the resulting aspect set depends only on the size of the set of primitives, not on the number of object models or on object model complexity. The primitives that we have selected are based on Biederman's geons (Biederman, 1985), offering a rich vocabulary with which to construct objects.

A qualitative recognition paradigm has advantages over and above that of restricting the number of aspects we require to represent our primitives. Such a system would be of great value in many robotic vision applications requiring object identification. For example, the sorting of distinct objects by a robot may require only that the objects be classified; position and pose determination may be unnecessary. Or, for an autonomous vehicle, quickly identifying the objects in the field of view may guide the vision system to select a course of action. For example, noticing a tree on the side of the road may not alarm the system, while

noticing another vehicle may invoke modules to estimate its velocity. Finally, a qualitative object recognition engine could provide a coarse front end to a more quantitative recognition engine; identification of the object's generic class could be used to invoke specific modules to distinguish among instances of subclasses. A coarse-to-fine approach to object recognition incurs the cost of extracting finer detail only when necessary.

In this chapter, we present only our object modeling paradigm; techniques for primitive extraction and model matching will not be presented. The second section discusses some of the issues in selecting an object modeling scheme and gives the motivation for our choice. The third section presents the object-centered component of our representation, while the fourth section presents the viewer-centered component. In the fifth section, we tie together the two representations with some probabilistic results based on an extensive analysis of the primitives over the viewing sphere; the sixth section evaluates the integrated representation. In the seventh and eighth sections we discuss related work and draw conclusions about this approach.

Object Modeling for 3-D Recognition

Many object modeling techniques have been applied to the task of 3-D object recognition (e.g., Requicha, 1980; Srihari, 1981; Binford, 1982; Besl and Jain, 1985; Chin and Dyer, 1986). In any object modeling scheme, an object is composed of one or more features or primitives; examples include lines, vertices, surface patches, generalized cylinders, and superquadrics. Models may be object-centered constructions of 3-D primitives, or viewer-centered constructions of 2-D primitives. In the former, the model is independent of viewpoint, while in the latter, each distinct view of the model generates a unique representation. When selecting a modeling technique for 3-D object recognition, a number of trade-offs must be considered. For example, complex primitives such as generalized cylinders are more difficult to extract from an image than simple primitives such as lines and curves. However, it is more efficient to search a database of objects, each composed of a relatively small number of complex primitives, than a database of objects, each composed of a relatively large number of simple primitives. Figure 13.1 illustrates the trade-offs in selecting a representation scheme.

Many approaches to 3-D object recognition (e.g., Lowe, 1985; Huttenlocher and Ullman, 1987; Thompson and Mundy, 1987; Lamdan et al., 1988) limit the bottom-up feature extraction process to 2-D primitives such as line segments, corners, zeroes of curvature, and 2-D perceptual structures. These features are appealing due to their viewpoint invariance; however, they suffer the shortcoming of requiring complex models. Since a model in these representations typically consists of a large number of very similar primitives, searching a large model database becomes inefficient. As a result of this limitation, these recognition systems have only been successfully applied to object databases containing one or two objects. In addition, the simplicity and 2-D nature of the indexing primitives

noticing another vehicle may invoke modules to estimate its velocity. Finally, a qualitative object recognition engine could provide a coarse front end to a more quantitative recognition engine; identification of the object's generic class could be used to invoke specific modules to distinguish among instances of subclasses. A coarse-to-fine approach to object recognition incurs the cost of extracting finer detail only when necessary.

In this chapter, we present only our object modeling paradigm; techniques for primitive extraction and model matching will not be presented. The second section discusses some of the issues in selecting an object modeling scheme and gives the motivation for our choice. The third section presents the object-centered component of our representation, while the fourth section presents the viewer-centered component. In the fifth section, we tie together the two representations with some probabilistic results based on an extensive analysis of the primitives over the viewing sphere; the sixth section evaluates the integrated representation. In the seventh and eighth sections we discuss related work and draw conclusions about this approach.

Object Modeling for 3-D Recognition

Many object modeling techniques have been applied to the task of 3-D object recognition (e.g., Requicha, 1980; Srihari, 1981; Binford, 1982; Besl and Jain, 1985; Chin and Dyer, 1986). In any object modeling scheme, an object is composed of one or more features or primitives; examples include lines, vertices, surface patches, generalized cylinders, and superquadrics. Models may be object-centered constructions of 3-D primitives, or viewer-centered constructions of 2-D primitives. In the former, the model is independent of viewpoint, while in the latter, each distinct view of the model generates a unique representation. When selecting a modeling technique for 3-D object recognition, a number of trade-offs must be considered. For example, complex primitives such as generalized cylinders are more difficult to extract from an image than simple primitives such as lines and curves. However, it is more efficient to search a database of objects, each composed of a relatively small number of complex primitives, than a database of objects, each composed of a relatively large number of simple primitives. Figure 13.1 illustrates the trade-offs in selecting a representation scheme.

Many approaches to 3-D object recognition (e.g., Lowe, 1985; Huttenlocher and Ullman, 1987; Thompson and Mundy, 1987; Lamdan et al., 1988) limit the bottom-up feature extraction process to 2-D primitives such as line segments, corners, zeroes of curvature, and 2-D perceptual structures. These features are appealing due to their viewpoint invariance; however, they suffer the shortcoming of requiring complex models. Since a model in these representations typically consists of a large number of very similar primitives, searching a large model database becomes inefficient. As a result of this limitation, these recognition systems have only been successfully applied to object databases containing one or two objects. In addition, the simplicity and 2-D nature of the indexing primitives

noticing another vehicle may invoke modules to estimate its velocity. Finally, a qualitative object recognition engine could provide a coarse front end to a more quantitative recognition engine; identification of the object's generic class could be used to invoke specific modules to distinguish among instances of subclasses. A coarse-to-fine approach to object recognition incurs the cost of extracting finer detail only when necessary.

In this chapter, we present only our object modeling paradigm; techniques for primitive extraction and model matching will not be presented. The second section discusses some of the issues in selecting an object modeling scheme and gives the motivation for our choice. The third section presents the object-centered component of our representation, while the fourth section presents the viewer-centered component. In the fifth section, we tie together the two representations with some probabilistic results based on an extensive analysis of the primitives over the viewing sphere; the sixth section evaluates the integrated representation. In the seventh and eighth sections we discuss related work and draw conclusions about this approach.

Object Modeling for 3-D Recognition

Many object modeling techniques have been applied to the task of 3-D object recognition (e.g., Requicha, 1980; Srihari, 1981; Binford, 1982; Besl and Jain, 1985; Chin and Dyer, 1986). In any object modeling scheme, an object is composed of one or more features or primitives; examples include lines, vertices, surface patches, generalized cylinders, and superquadrics. Models may be object-centered constructions of 3-D primitives, or viewer-centered constructions of 2-D primitives. In the former, the model is independent of viewpoint, while in the latter, each distinct view of the model generates a unique representation. When selecting a modeling technique for 3-D object recognition, a number of trade-offs must be considered. For example, complex primitives such as generalized cylinders are more difficult to extract from an image than simple primitives such as lines and curves. However, it is more efficient to search a database of objects, each composed of a relatively small number of complex primitives, than a database of objects, each composed of a relatively large number of simple primitives. Figure 13.1 illustrates the trade-offs in selecting a representation scheme.

Many approaches to 3-D object recognition (e.g., Lowe, 1985; Huttenlocher and Ullman, 1987; Thompson and Mundy, 1987; Lamdan et al., 1988) limit the bottom-up feature extraction process to 2-D primitives such as line segments, corners, zeroes of curvature, and 2-D perceptual structures. These features are appealing due to their viewpoint invariance; however, they suffer the shortcoming of requiring complex models. Since a model in these representations typically consists of a large number of very similar primitives, searching a large model database becomes inefficient. As a result of this limitation, these recognition systems have only been successfully applied to object databases containing one or two objects. In addition, the simplicity and 2-D nature of the indexing primitives

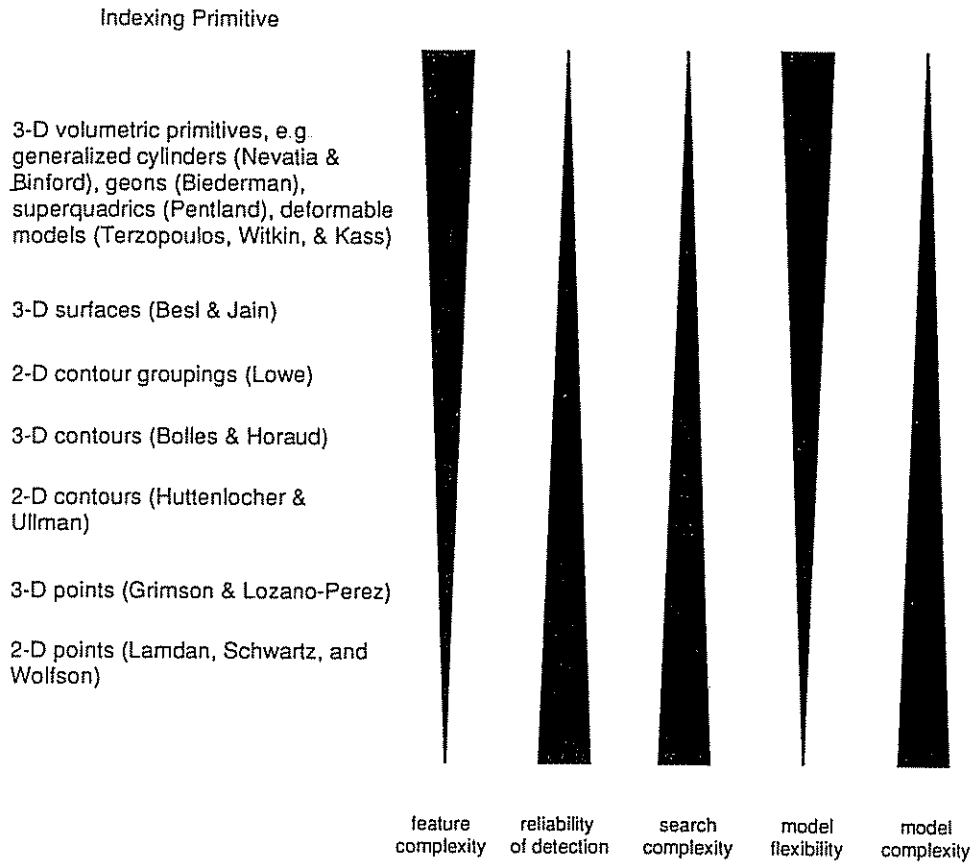


FIGURE 13.1. The trade-offs in choosing modeling primitives.

require a complex verification procedure involving the determination of the object's pose with respect to the image. Indexing with weak hypotheses shifts the burden of recognition to verification, resulting in a top-down system. Our approach is to extend the bottom-up process to the point of inferring 3-D volumetric primitives with which we index into the database. Since our primitives capture more information than simple point or line primitives, they provide more discriminating indices to less complex models. These higher order indices provide a foundation for a more bottom-up unexpected-object recognition system (Rosenfeld, 1986).

A 3-D recognition system based on the bottom-up extraction of 3-D volumetric indexing primitives raises the obvious question: How do we extract the primitives from the image? To meet this requirement, we employ a viewer-centered aspect representation to model an object's primitives. This differs from traditional aspect-based recognition systems where the entire object is modeled as a set of aspects (e.g., Chakravarty and Freeman, 1982; Ikeuchi and Kanade, 1988). The

advantage of aspect representations is that they reduce the 3-D matching proble to a 2-D matching problem; each 2-D aspect defines the object's identity and 3- pose. The disadvantage of aspect representations is that they incur a high cost matching as each 3-D object must be represented by many different view Moreover, as the complexity of an object increases, so do the number of vie required to represent it. To restrict the number of aspects representing the mod primitives, we model primitive classes rather than primitive instances. Thus, f each primitive, we define a set of aspects that is invariant to minor changes primitive shape. More specifically, we define our aspects to be invariant changes in line length, curvature, and angle. Consequently, a primitive ma undergo changes in surface dimension and surface curvature without introducir new aspects. This constrains our primitives to possess a qualitative nature, ca turing only the gross shape characteristics of the object.

The integration of object-centered and viewer-centered models combines th advantages of each scheme while avoiding their disadvantages. The extraction 3-D volumetric primitives provides a highly selective index into a database compact object representations. Extraction of the primitives from the image performed by matching image contours to the contours comprising a set of 2- aspects; pose determination is inherent in the 2-D matching process. To restri the explosion of primitive aspects, we represent each primitive class, rather tha each primitive instance, with a set of aspects. Unlike traditional aspect-based re ognition systems, the resulting number of aspects is constant and independent the size of the model database. In the following sections, we discuss the represe tation in more detail.

The Object-Centered Modeling Component

The goal of the object-centered modeling component is to define a set of thre dimensional volumetric primitives that, when assembled together, comprise large set of concrete objects in the world. The constraints on these primitiv are two-fold: they must be rich enough to describe real objects, yet simple enou to be reliably extracted from a contour image. The primitives, in turn, wi be mapped into a set of viewer-centered aspects. Any selection of modelin primitives would support our approach; however, we seek a set of primitiv whose aspect set will remain stable under minor changes in primitive shap For example, the aspects should be invariant to changes in the primitive's scal dimensions, and curvature (if the primitive is curved). Otherwise, the resul ing large aspect set introduces the same matching inefficiency as tradition aspect-based recognition systems. To meet these requirements, we have chose an object representation based on Biederman's Recognition by Componen (RBC) theory (Biederman, 1985). RBC suggests that from nonaccidental rel tions in the image, a set of contrastive dichotomous (e.g., straight vs. curved axi and trichotomous (e.g., constant vs. tapering vs. expanding/contracting cros sectional sweep) 3-D primitive properties can be determined. The values of the

properties give rise to a set of 36 shapes, called geons. Biederman claims that these geons constitute a rich set of primitive volumetric components that, when assembled together, can be used to model real world objects for the purpose of fast object recognition.

Many 3-D object recognition systems employ 3-D volumetric primitives to construct objects. Biederman's geons are a restricted class of generalized cylinders (e.g., Binford, 1971; Agin and Binford, 1976; Nevatia and Binford, 1977; Brooks, 1983) whose cross-section, axis, and sweep properties are arbitrary functions. Superquadrics (Gardiner, 1965) provide a volumetric representation requiring fewer parameters than generalized cylinders. Pentland (1986) first applied superquadrics to primitive modeling for object recognition, while Pentland (1987a) and Solina (1987) have achieved considerable success in deriving superquadric primitives from range data. Terzopoulos et al. (1987) propose symmetry-seeking deformable 3-D shape models, which they have successfully applied to the recovery of 3-D shape and nonrigid motion from natural imagery (Terzopoulos et al., 1988). Although generalized cylinders, superquadrics, and active models provide a rich language for describing parts, their extraction from the image is computationally complex. Biederman's geons, requiring only a few parameters, are an appropriate selection for the purposes of qualitative object modeling.

The Primitives

A generalized cylinder is defined by a cross-section function, an axis function, and a sweeping function; its shape results from sweeping the cross-section along the axis. Biederman (1985) mapped these three continuous functions to dichotomous and trichotomous properties. His 36 geons result from the Cartesian product of the possible values of these properties, which are defined as follows:

1. **Cross-Section Shape:** The cross-section shape can be either straight edged or curved edged.
2. **Cross-Section Symmetry:** The cross-section shape can be either rotationally symmetric, reflectively symmetric, or asymmetric.
3. **Axis Shape:** The axis can be either straight or curved.
4. **Cross-Section Sweep:** The cross-section can either remain constant, increase in size, or increase and then decrease in size as it is swept along the axis.

As a basis for initial investigation, we have defined a set of 10 primitives representing a restricted subset of Biederman's geons:

1. rectangular cross-section, straight axis, and constant cross-section size
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 nor ending with a point
9. elliptical cross-section, straight axis, and ellipsoidally increasing then decreasing cross-section size, starting and ending with a point
10. elliptical cross-section, curved axis, and constant cross-section size

Our three-property characterization resembles Biederman's four-property taxonomy; however, we have imposed additional restrictions in an effort to reduce the number of aspects and simplify the investigation. Nevertheless, the above primitive set forms a basis from which we can model a significant number of objects. The 10 primitives have been modeled using Pentland's SuperSketch 3-D modeling tool (Pentland, 1987b), and are illustrated in Figure 13.2. More primitives can easily be added to enrich the vocabulary.

Primitive Attachment

Having defined a set of modeling primitives, we must decide how to connect them to construct objects. We adopt a convention based on a labeling of each primitive's attachment surfaces. For 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 10 primitives can be found in Dickinson et al. (1989). We restrict any junction of two primitives to involve exactly one attachment surface from each primitive. Figure 13.3 presents an example object and its representation.

Both the primitive description and the interconnection description have been oversimplified to demonstrate the approach. Many enhancements are possible that would provide a much richer vocabulary for describing objects. For example, although not viewpoint invariant, additional properties such as cross-section extent, axis extent, and axis curvature provide important cues for recognition. Although these properties are quantitative, we could treat them as symbolic, based on a qualitative partitioning of the property range. For example, an axis might be "slightly curved" or "strongly curved" depending on its average curvature value.

In addition to specifying the two surfaces participating in the junction of two primitives, we could specify the position of the join on each surface. For example, a primitive attached to a rectangular planar surface may be attached near the middle, the sides, the corners, or the ends of the surface. A primitive attached to

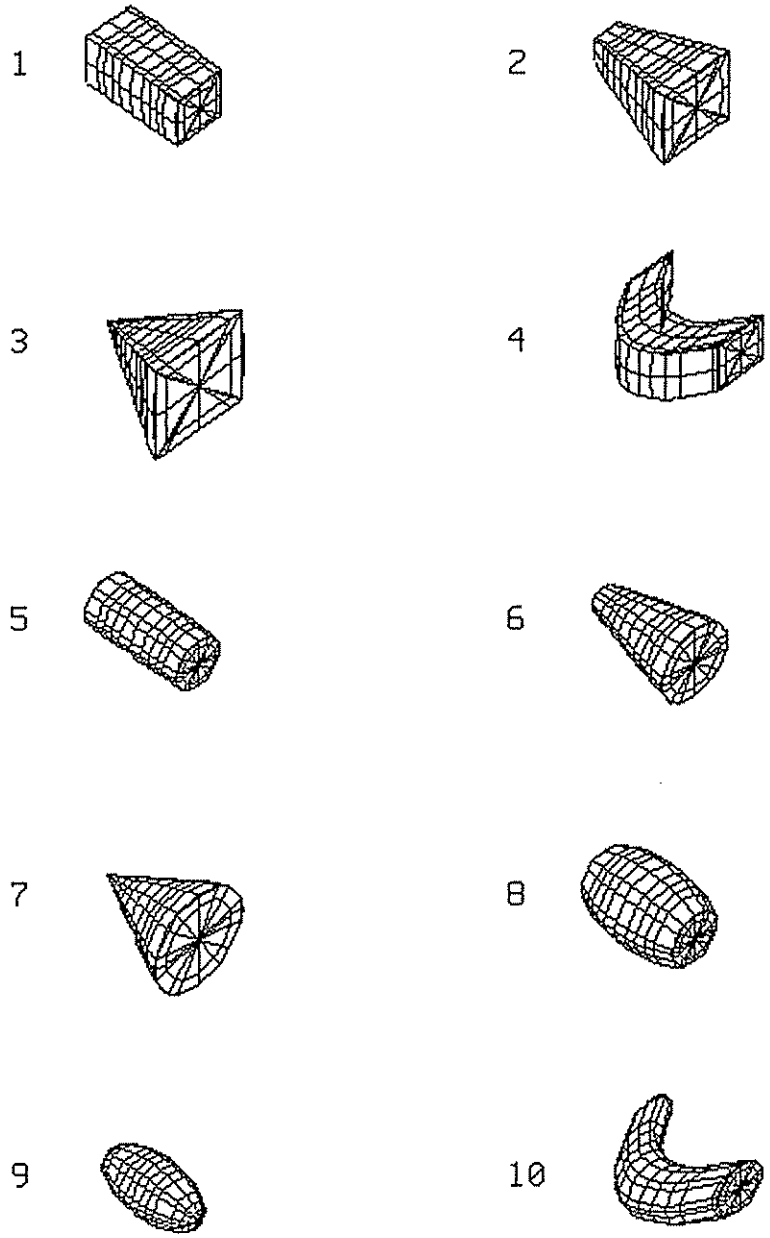


FIGURE 13.2. The 10 object modeling primitives.

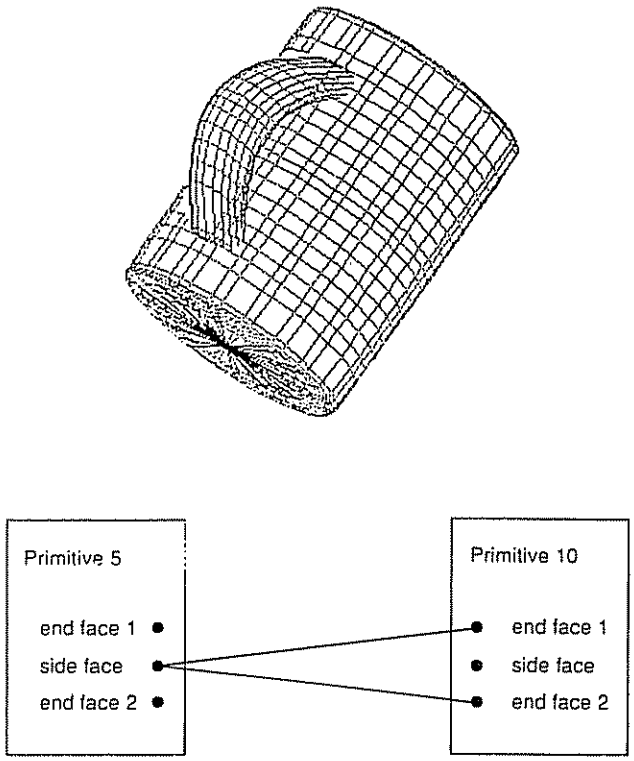


FIGURE 13.3. Example object and its representation.

an ellipsoidal primitive may be attached near the middle, the sides, or the ends of the ellipsoid. Again, we seek a qualitative localization of interconnection; we want to avoid an exact quantitative specification. Another enhancement to our vocabulary would be to describe the relative sizes of the primitives and the angles at which they join. For example, relative size measures such as "much larger than," "slightly larger than," or "roughly equal," and join angles such as "acute" or "perpendicular," are additional interprimitive relations that enhance the description of the object. The resolution of these descriptors would depend on the similarity of the objects in the database; perhaps having both coarse and fine descriptors would maximize matching efficiency.

Viewer-Centered Modeling Component

For each of the 10 primitive classes, we define a set of 2-D characteristic views, or aspects. Each aspect represents a set of topologically equivalent views of the primitive. To extract instances of the primitives from the image, we match image

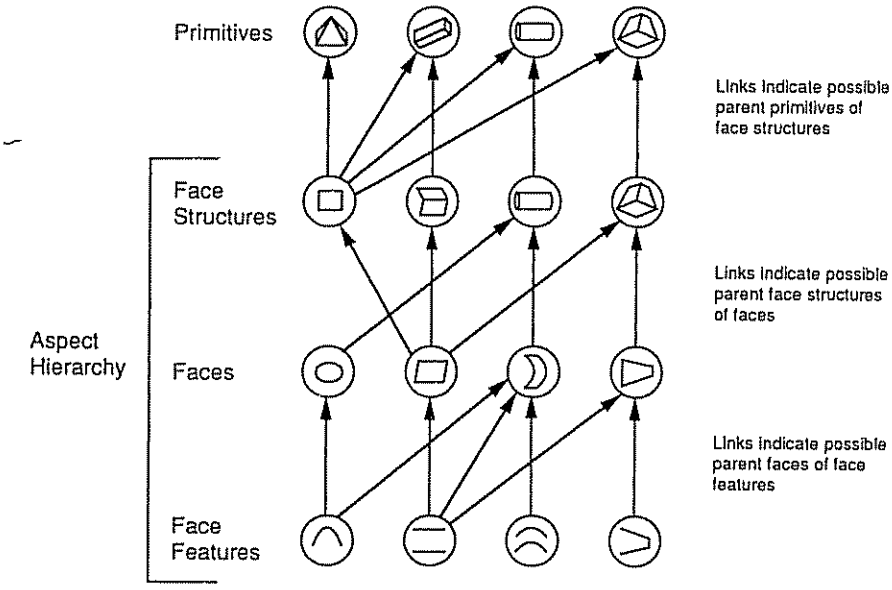


FIGURE 13.4. The aspect hierarchy.

contours against the set of aspects; a match not only identifies the primitive but qualitatively specifies its orientation. Unfortunately, if a primitive is occluded from a given 3-D viewpoint, its projected aspect in the image will also be occluded. In addition, the intersection of two primitives may alter the projected aspect of either primitive. To accommodate the matching of partial aspects to the set of aspects, we introduce a representation called the aspect hierarchy.

The aspect hierarchy consists of three levels, based on the faces appearing in the aspect set. A face is defined to be a closed cycle of image contours, e.g., a polygon, containing no other cycles. At the top level of the aspect hierarchy, we have the set of aspects, which we call face structures. Ideally, we would like to match image contours directly to the face structures. However, due to occlusion, it is unlikely that complete face structures will be visible. Some component faces of a face structure may be completely occluded, others partially occluded. The set of component faces of all face structures represents the middle level of the aspect hierarchy. Hence, we reduce face structure extraction to face extraction. However, we again run into the problem of occlusion, resulting in faces appearing in the image that are not included in the face level. Nevertheless, there may be subsets of face contours that survive occlusion and offer a mechanism for matching. Thus, at the bottom level of the aspect hierarchy, we have the face features that comprise the set of faces. Figure 13.4 illustrates a portion of the aspect hierarchy, while the following subsections describe the levels of the aspect hierarchy in more detail.

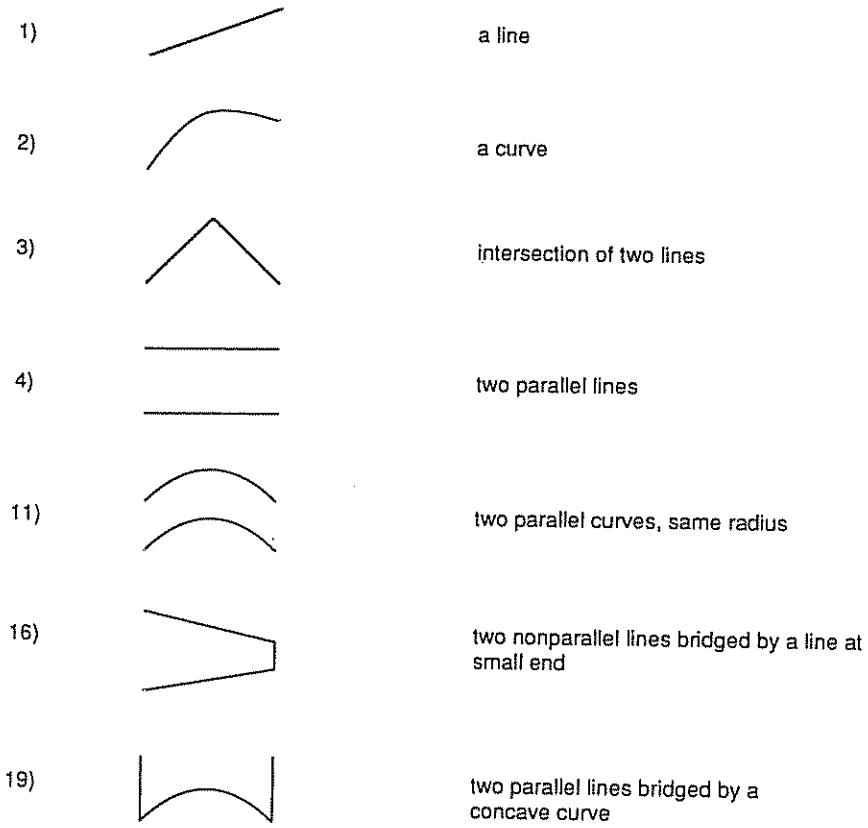


FIGURE 13.5. Examples of face features.

Face Features

The face features represent all subsets of lines and curves comprising the faces. Figure 13.5 illustrates a few of the 31 face features based on our 10 primitives; the complete set of face features can be found in Dickinson et al. (1989). The relations between face feature components (lines and curves) represent nonaccidental properties of lines including parallelism, symmetry, and cotermination. These relations, described by Biederman (1985) as a basis for the extraction of geons, are a subset of the nonaccidental properties suggested by Lowe (1985) and Witkin and Tenenbaum (1983). The important characteristic of the face features is that they represent *qualitative* relationships among *qualitative* lines; exact lengths of lines, distances between lines, angles between lines, curvature, etc., are not represented.

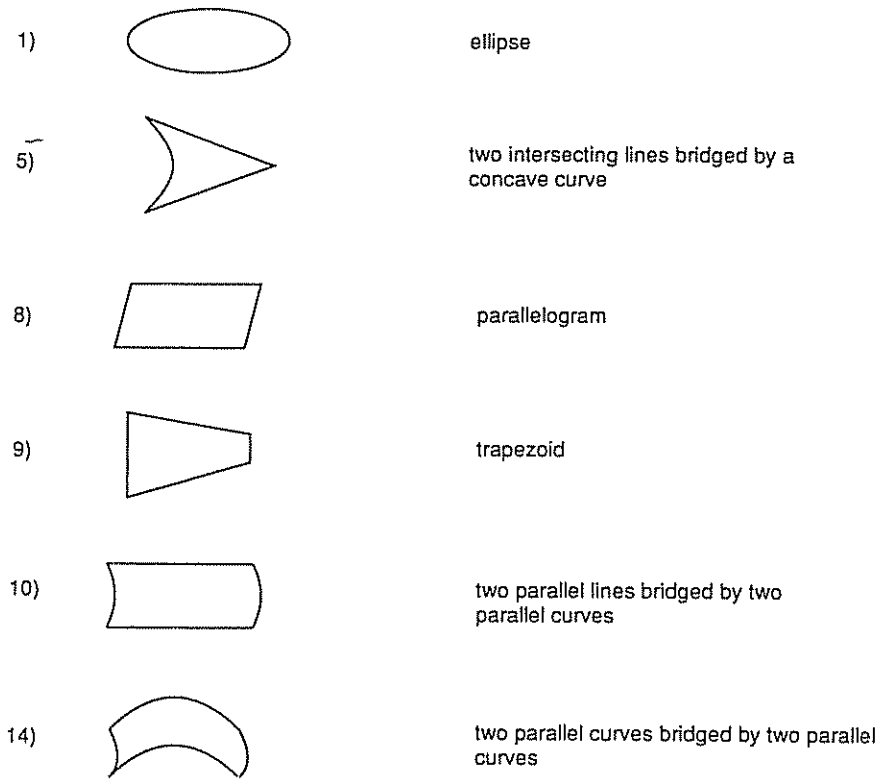


FIGURE 13.6. Examples of faces.

Faces

The faces represent the set of polygons appearing in the aspects. Figure 13.6 illustrates a few examples of our 16 faces; the complete set of faces can be found in Dickinson et al. (1989). As mentioned earlier, the faces form the backbone of the aspect hierarchy. Each differs in the number of constituent lines, the types of lines, or the nonaccidental relations between the lines. Since a face definition is invariant to changes in constituent line length and angle (provided the defining line relationships still hold), each face in Figure 13.6 represents only one of many possible instances defining the class.

Face Structures

The face structures represent connected sets of faces; each face in the structure shares a line with at least one other face in the structure. Figure 13.7 illustrates

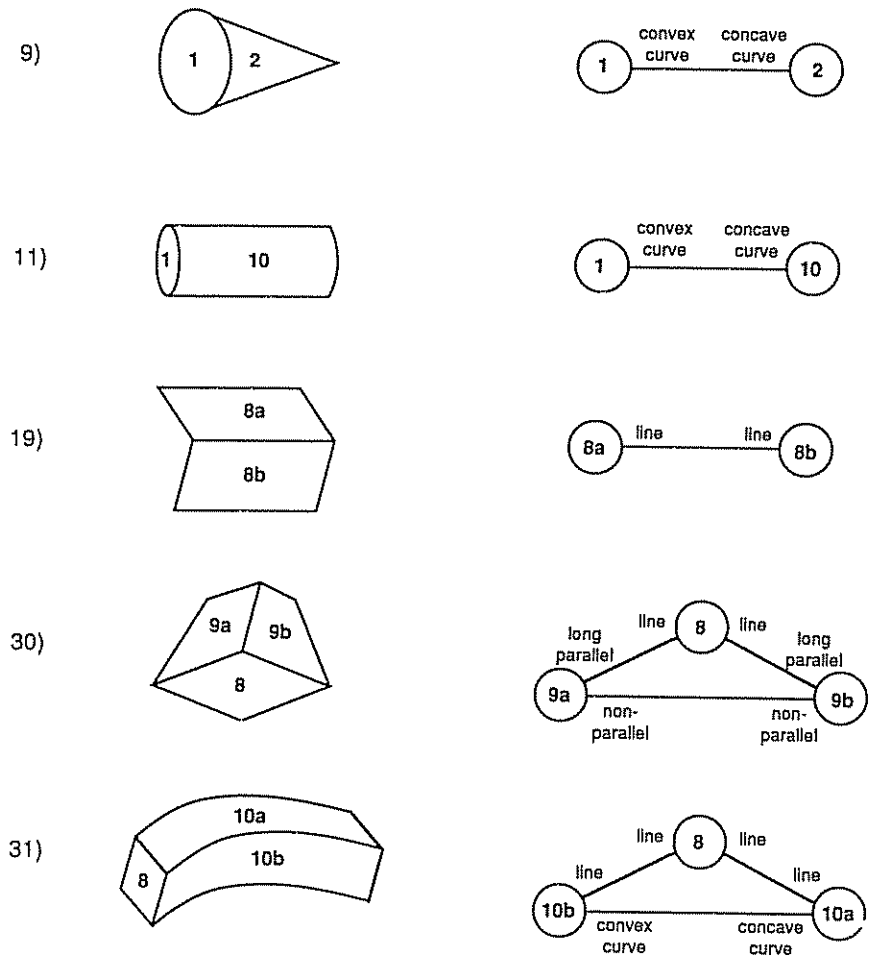


FIGURE 13.7. Examples of face structures.

a few examples of our 37 face structures; the complete set of face structures can be found in Dickinson et al. (1989). A face structure can be represented by a connected graph, with the nodes representing faces and the arcs representing the sharing of lines between faces; arc labels indicate which line is being shared.

Combining the Two Components

A given face feature may be common to a number of faces. Similarly, a given face may be a component of a number of face structures, while a given face structure may be the projection of a number of primitives. To capture these ambiguities, a matrix maps face features to faces, while another matrix maps faces to face struc-

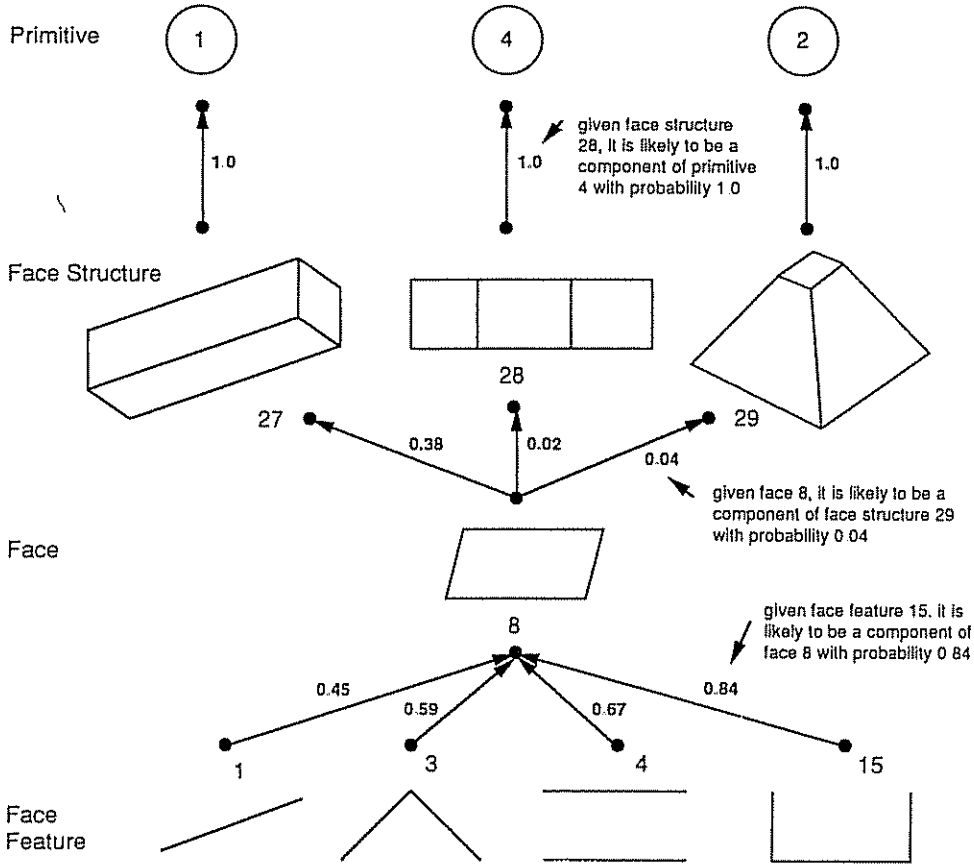


FIGURE 13.8. Combining the object-centered and viewer-centered models.

tures. To tie together the object-centered and viewer-centered representations, we define a third matrix mapping the top level of the aspect hierarchy, the face structure level, to the primitives.

In many cases, a feature (face feature, face, or face structure) could be a component of more than one parent feature at the next higher level; however, some parents might be more likely than others. The entries in the three matrices capture this likelihood. For example, consider the matrix mapping faces to face structures; the rows represent faces while the columns represent face structures. If a particular face can be a component of 10 different face structures, then those 10 column entries corresponding to the 10 face structures contain a value from 0 to 1.0, indicating the probability that the face is part of that particular face structure. Thus, the entries along each row sum to 1.0. Figure 13.8 presents a portion of the aspect hierarchy and related primitives along with the corresponding portions of the matrices.

TABLE 13.1. Superquadric definitions of the 10 primitives.

	Primitive									
Parameter Value	1	2	3	4	5	6	7	8	9	10
x-size	15	15	15	15	15	15	15	15	15	15
y-size	15	15	15	15	15	15	15	15	15	15
z-size	30	30	30	30	30	30	30	30	30	30
ϵ_1	.05	.05	.05	.05	1.0	1.0	1.0	1.0	1.0	1.0
ϵ_2	.05	.05	.05	.05	.05	.05	.05	.05	1.0	.05
z-axis bend	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
z-axis taper	0.0	0.5	1.0	0.0	0.0	0.5	1.0	0.0	0.0	0.0
z-axis pinch	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0

To generate the probabilities in the tables mapping face features to faces, faces to face structures, and face structures to primitives, we first modeled our 3-D volumetric primitives using the SuperSketch modeling tool (Pentland, 1987b). SuperSketch models each primitive with a superquadric surface subject to deformation. The superquadric with length, width, and breadth a_1, a_2, and a_3 is described (adopting the notation $\cos\eta = \mathbf{C}_\eta$, $\sin\omega = \mathbf{S}_\omega$) by the following equation:

$$\mathbf{X}(\eta,\omega) = \begin{pmatrix} a_1\mathbf{C}_\eta^{\epsilon 1}\mathbf{C}_\omega^{\epsilon 2} \\ a_2\mathbf{C}_\eta^{\epsilon 1}\mathbf{S}_\omega^{\epsilon 2} \\ a_3\mathbf{S}_\eta^{\epsilon 1} \end{pmatrix}$$

where $\mathbf{X}(\eta,\omega)$ is a three-dimensional vector that sweeps out a surface parameterized in latitude η and longitude ω, with the surface's shape controlled by the parameters ε_1 and ε_2. The superquadric can be deformed by stretching, bending, twisting, or tapering. The SuperSketch superquadric definitions for the 10 primitives are given in Table 13.1.

The next step in generating the probability tables involves rotating each superquadric primitive about its internal x, y, and z axes in 10° intervals. The resulting quantization of the viewing sphere gives rise to 648 different views per superquadric primitive. However, we can exploit symmetries of the primitives to significantly reduce the number of views (688 views for all primitives). For each view, we orthographically project the superquadric primitive into the image plane. The final step involves a manual analysis of the images, noting each feature (face feature, face, and face structure) and its parent. The resulting frequency distribution gives rise to the three probability matrices given in Dickinson et al. (1989).

It should be emphasized that the results offer only a crude approximation to the true probabilities. A more thorough analysis would vary the dimensions, curvature, expansion rate, etc. of the primitives at a finer resolution on the viewing sphere. The resulting explosion of views would require an automated tool to

perform the analysis and generate the probabilities. However, we believe that the probabilities will not change significantly in a more thorough analysis.

The three matrices can be used to guide the process of extracting the primitives from image contours. For example, if a face extracted from the image fails to match one of the 16 face types due to occlusion, the matrix mapping face features to faces can predict the most likely face type given a face feature belonging to the face. Once a face has been identified, the matrix mapping faces to face structures can predict the most likely face structure containing that face. Finally, given a face structure in the image, the matrix mapping face structures to primitives can predict the most likely primitive to which the face structure belongs. At each level, the matrices provide a heuristic to guide the search through the interpretations. The details of the processes used in both primitive extraction and model matching will not be presented in this chapter.

Evaluating the Aspect Hierarchy

An alternative approach to our hierarchical aspect representation would be to map the features at the lowest level (in our case, the face features) directly to the 3-D models (in our case, the 3-D primitives), an approach advocated by Lowe (1985), Huttenlocher and Ullman (1987), and Lamdan et al. (1988). In such a scenario, a given face feature index would return a set of candidate primitives containing that face feature. This approach has several drawbacks. First, the complexity of the primitives would increase to accommodate constituent face features. We would also face the problem that a weak hypothesis based on simple features requires a top-down verification step; the more qualitative the primitive, the more difficult the verification. Finally, simple indexing features do not provide strong orientation constraints on the primitives. For example, a pair of intersecting lines may lead to primitive hypotheses in many different orientations, whereas the face structure encompassing the lines constrains the orientation. In fact, we use the face structure label as a qualitative specification of a primitive's orientation.

In addition to the above heuristic arguments, we can make a quantitative case for the aspect hierarchy based on the three mapping matrices. By multiplying together matrices representing adjacent levels of the aspect hierarchy, we can generate new matrices mapping face features to face structures, faces to primitives, and face features to primitives. The results can best be seen in a set of histograms. To generate the histogram between two levels in the aspect hierarchy, we retain only the strongest probability arc emanating from each feature at the lower level. This indicates the degree of ambiguity in the mapping. For example, a node having two emanating arcs with values 0.50 and 0.50 is clearly inferior to a node having three emanating arcs with values 0.90, 0.05, and 0.05. Making an inference at the node with two emanating, equal probability arcs is a 50–50 guess, whereas there is a clear choice at the node with three emanating arcs. Clearly, having fewer emanating arcs is not as important as having a distinctly high prob-

ability arc. Once the strongest probability arc emanating from each node has been retained, we simply count the number of remaining arcs falling in each of 10 probability intervals. The resulting histograms (percentage of nodes whose emanating highest probability arc falls within given probability range) are presented in Figure 13.9.

Working backwards from the primitives, we can compare mappings from the face structures, faces, and face features to the primitives; the three histograms are illustrated in Figure 13.9c, e, and f, respectively. The face structure to primitive mapping is the strongest, with 90% of the face structure nodes having a high probability (0.80–1.0) arc. At the face structure level, we can compare mappings from the faces and face features; the two histograms are illustrated in Figure 13.9b and d, respectively. In this case, the mapping from the faces is much less ambiguous than the mapping from the face features. The final mapping from face features to faces is illustrated in Figure 13.9a.

The aspect hierarchy effectively prunes the mapping from face features to primitives by introducing intermediate constraints in the form of faces and face structures. The histograms suggest that for a typical set of 3-D modeling primitives, image regions, or faces, are the most appropriate features for recognition. Moreover, these faces should be grouped into the more complex face structures, providing a less ambiguous mapping to the primitives and further constraining their orientation. Only when a face's shape is altered due to primitive occlusion should we descend to the face feature level.

Related Work

Brooks' ACRONYM system (Brooks, 1983) 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 the subclasses of a particular airplane, while our system cannot; however, in distinguishing an airplane from, say, a horse, we avoid detailed quantitative constraints.

In contrast to ACRONYM's top-down approach, Lowe's SCERPO system (Lowe, 1985) 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

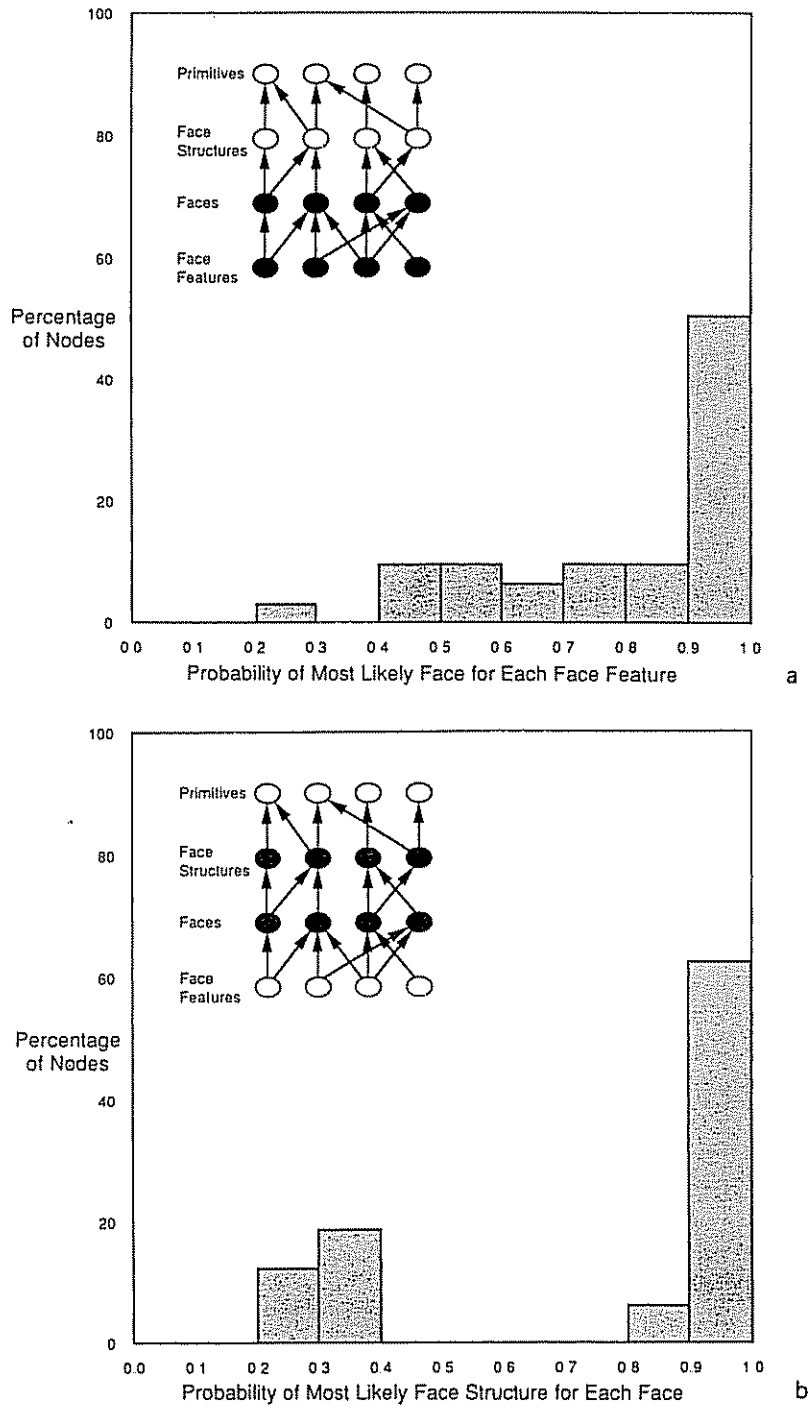
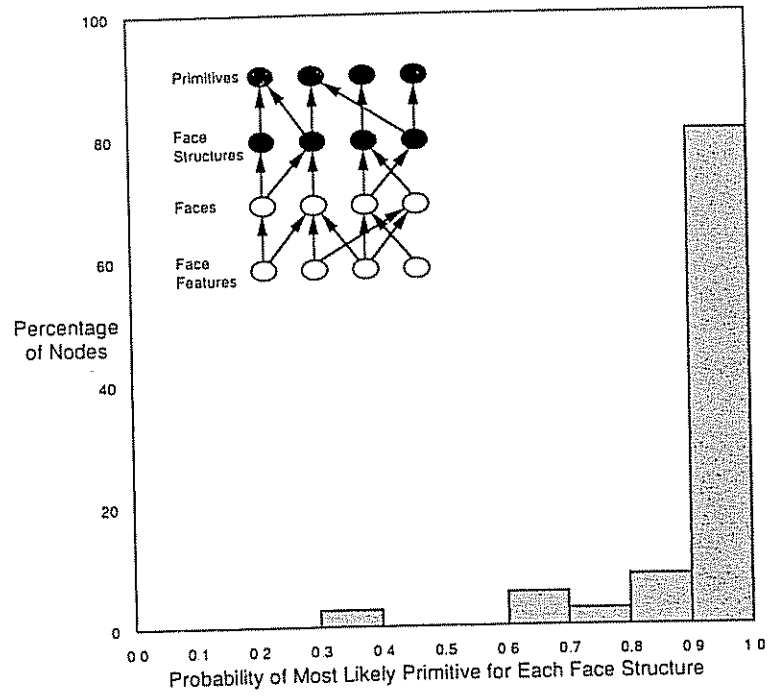
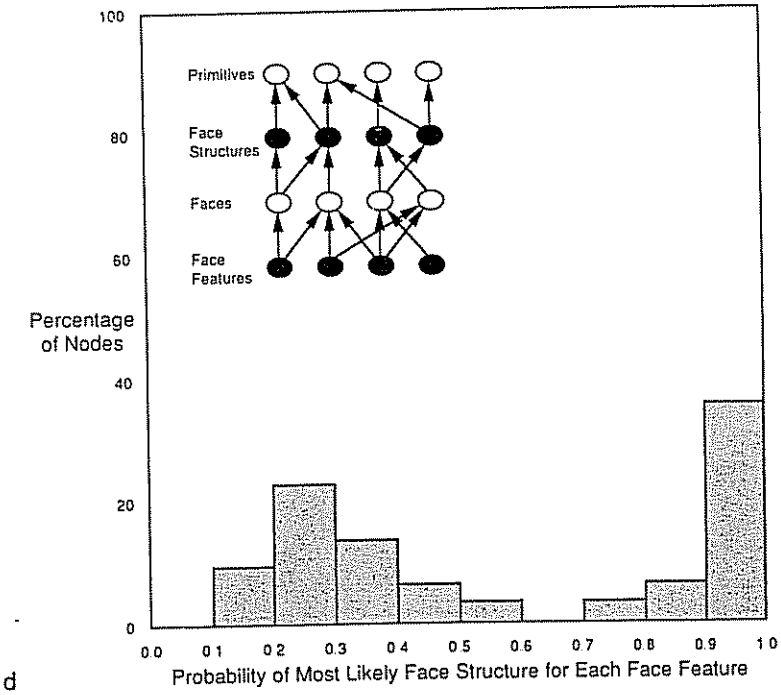


FIGURE 13.9. (a) Face feature to face mapping. (b) Fact to face structure mapping.

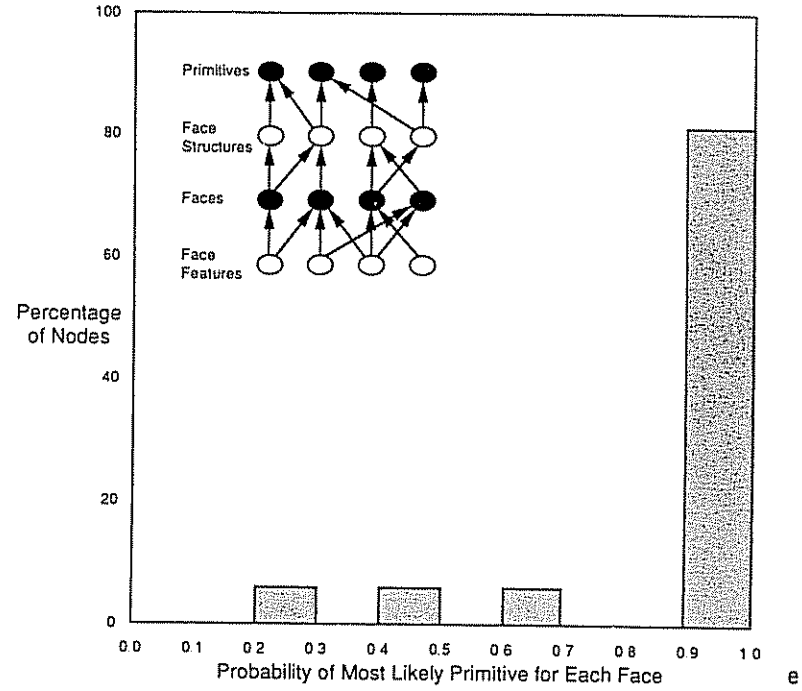


c



d

FIGURE 13.9. (c) Face structure to primitive mapping. (d) Face feature to face structure mapping.



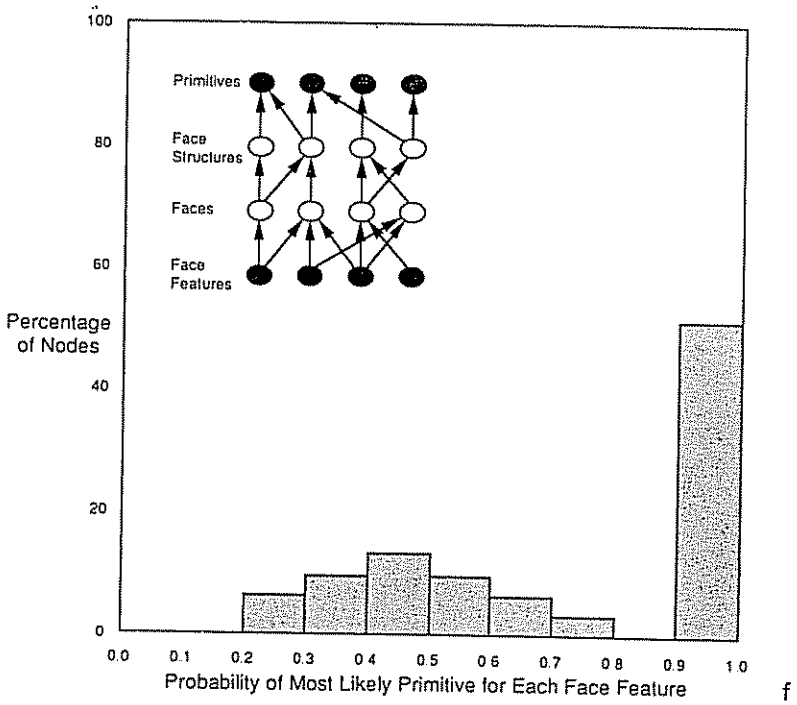


FIGURE 13.9. (e) Face to primitive mapping. (f) Face feature to primitive mapping.

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 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. (1984) 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 to 3-D part instances in the model database, 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 (1985) proposed a set of primitives, called geons, based on the dichotomous and trichotomous properties of generalized cylinders. However, he failed to demonstrate how they may be extracted from the image, nor did he propose a control strategy for matching image features to models. Bergevin and Levine (1988a,b) 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 tangent 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. The technique assumes that the segmentation points can be paired. In addition, PARVO 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. Perhaps the greatest 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 (1982), and more recently by Ikeuchi and Kanade (1988). However, in these systems, the whole object is represented by the set of aspects. Thus, as the complexity of the object increases, so does the number of distinct aspects; automatically generating the distinguishable aspects is a difficult task. In our system, the aspects of a set of common parts or primitives have been generated and analyzed, and will be applicable to any objects constructed with these parts. Rather than matching against a large number of complex aspects, we plan to 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.

Conclusions

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, or curvature points. Unfortunately, these primitives do not provide very discriminating indices into a large database. In fact, there may be many instances of such primitives in just one model, resulting in many hypothesized matches. The resulting systems are very top-down or model driven in nature. To achieve a more bottom-up recognition system requires that we index into the model database with more discriminating, higher order primitives. However, the more complex the indexing primitive, the more difficult the primitive extraction.

We propose a representation integrating constructions of 3-D volumetric modeling primitives at the database level with a set of aspects that describes the primitives at the image level. To reduce the number of aspects, our primitives are qualitative in nature, with the set of primitive aspects invariant to minor changes in primitive shape. The resulting integration of object-centered and viewer-centered models provides the foundation for a more bottom-up unexpected-object recognition system. The qualitative nature of the representation is ideal for qualitative recognition, and could provide a coarse front end for a more quantitative recognition system. Although we demonstrate our approach using a particular choice of 3-D primitives, the integration of object-centered and viewer-centered representations using a probabilistic aspect hierarchy is equally applicable to any representation scheme modeling objects as constructions of 3-D volumetric primitives.

Acknowledgments. The authors would like to thank Suzanne Stevenson, Larry Davis, and Peter Cucka for insightful discussions and for their comments on earlier drafts of this chapter.

References

Agin GJ, Binford TO (1976). Computer description of curved objects. IEEE Transact. Comp. C-25(4):439–449.

Bergevin R, Levine MD (1988a). Recognition of 3-D objects in 2-D line drawings: An approach based on geons. Technical Report TR-CIM-88-24, McGill Research Centre for Intelligent Machines, McGill University.

Bergevin R, Levine MD (1988b). Hierarchical decomposition of objects in line drawings. Technical Report TR-CIM-88-25, McGill Research Centre for Intelligent Machines, McGill University.

Besl PJ, Jain RC (1985). Three-dimensional object recognition. ACM Comp. Surveys 17(1):75–145.

Biederman I (1985). Human image understanding: Recent research and a theory. Comp Vision, Graphics, Image Process. 32:29–73.

Binford TO (1971). Visual perception by computer. Proc. IEEE Conf. Systems Contro Miami, FL.

Binford TO (1982). Survey of model-based image analysis systems. Int. J. Robotics Re 1(1):18–64.

Bolles RC, Horaud P (1986). 3DPO: A three-dimensional part orientation system. Int. Robotics 5(3):3–26.

Brooks RA (1983). Model-based 3-D interpretations of 2-D images. IEEE Transact. Pa tern Anal. Machine Intell. 5(2):140–150.

Chakravarty I, Freeman H (1982). Characteristic views as a basic for three-dimension object recognition. Proc. SPIE Conf. Robot Vision, Arlington VA, 37–45.

Chin RT, Dyer CR (1986). Model-based recognition in robot vision. ACM Comp. Surve 18(1):67–108.

Dickinson SJ, Pentland AP, Rosenfeld A (1989). A representation for qualitative 3- object recognition integrating object-centered and viewer-centered models. Technic Report CAR-TR-453, Computer Vision Laboratory, Center for Automation Researcl University of Maryland.

Gardiner M (1965). The superellipse: A curve that lies between the ellipse and the recta gle. Sci. Am. 213:222–234.

Grimson WEL, Lozano-Perez T (1984). Model-based recognition and localization fro sparse range or tactile data. Int. J. Robotics Res. 3(3):3–35.

Huttenlocher DP, Ullman S (1987). Object recognition using alignment. Proc. First In Conf. Comp. Vision, London, 102–111.

Ikeuchi K, Kanade T (1988). Automatic generation of object recognition programs. Pro IEEE 76(8):1016–1035.

Lamdan Y, Schwartz JT, Wolfson HJ (1988). On recognition of 3-D objects from 2- images. Proc. IEEE Int. Conf. Robotics Automation, 1407–1413.

Lowe DG (1985). *Perceptual Organization and Visual Recognition*. Kluwer Academ Publishers. Norwell, Massachusetts.

Mulgaonkar PG, Shapiro LG, Haralick RM (1984). Matching 'sticks, plates and blol objects using geometric and relational constraints. Image Vision Comp. 2(2):85–98.

Nevatia R, Binford TO (1977). Description and recognition of curved objects. Arti Intell. 8:77–98.

Pentland AP (1986). Perceptual organization and the representation of natural forr Artif. Intell. 28:293–331.

Pentland AP (1987a). Recognition by parts. Proc. First Int. Conf. Comp. Vision, Londo 612–620.

Pentland AP (1897b). Towards an ideal 3-D CAD system. SPIE Conf. Machine Vision a Man-Machine Interface, San Diego.

Requicha AAG (1980). Representations for rigid solids. ACM Comp. Surveys 12(4 437–464.

Rosenfeld A (1987). Recognizing unexpected objects: A proposed approach. Proc. 19 DARPA Image Understanding Workshop, Los Angeles, 620–627.

Solina F (1987). Shape recovery and segmentation with deformable part models. Tec Report MS-CIS-87-111, GRASP LAB 128, University of Pennsylvania, Philadelphia, P.

Srihari SN (1981). Representation of three-dimensional digital images. ACM Comp. Sı veys 13(4):399–424.

Terzopoulos D, Witkin A, Kass M (1987) Symmetry-seeking models and 3D object reconstruction. Int. J. Comp. Vision 1:211–221.

Terzopoulos D, Witkin A, Kass M (1988). Constraints on deformable models: Recovering 3D shape and nonrigid motion. Artif. Intell. 36:91–123.

Thompson DW, Mundy JL (1987). Model-directed object recognition on the connection machine. Proc. 1987 DARPA Image Understanding Workshop, Los Angeles, 93–106.

Witkin AP, Tenenbaum JM (1983). On the role of structure in vision. In *Human and Machine Vision*, (J Beck, B Hope, A Rosenfeld, eds.), pp 481–543. Academic Press, New York.