International Journal of Computer Vision 19(2), 147–167 (1996)

Recognition by Prototypes

RONEN BASRI
Dept. of Applied Math, The Weizmann Institute of Science, Rehovot 76100, Israel
ronen@wisdom.weizmann.ac.il

Received June 21, 1993; Revised November 2, 1994 and June 2, 1995

Abstract. A scheme for recognizing 3D objects from single 2D images under orthographic projection is introduced. The scheme proceeds in two stages. In the first stage, the *categorization stage*, the image is compared to prototype objects. For each prototype, the view that most resembles the image is recovered, and, if the view is found to be similar to the image, the class identity of the object is determined. In the second stage, the *identification stage*, the observed object is compared to the individual models of its class, where classes are expected to contain objects with relatively similar shapes. For each model, a view that matches the image is sought. If such a view is found, the object's specific identity is determined. The advantage of categorizing the object before it is identified is twofold. First, the image is compared to a smaller number of models, since only models that belong to the object's class need to be considered. Second, the cost of comparing the image to each model in a class is very low, because correspondence is computed once for the whole class. More specifically, the correspondence and object pose computed in the categorization stage to align the prototype with the image are reused in the identification stage to align the individual models with the image. As a result, identification is reduced to a series of simple template comparisons. The paper concludes with an algorithm for constructing optimal prototypes for classes of objects.

1. Introduction

Recognition is a task of identifying portions of the image with object models stored in memory. One difficulty in recognition is that objects appear different from different viewpoints. Model-based approaches to recognition usually handle this problem by recovering the position and orientation (*pose*) of the object in the image and bringing the model to the recovered pose (Fischler and Bolles, 1981; Lowe, 1985; Faugeras and Hebert, 1986; Chien and Aggarwal, 1987; Thompson and Mundy, 1987; Ullman, 1989; Huttenlocher and Ullman, 1990; Basri and Ullman, 1993). This approach involves time-consuming algorithms requiring, for instance, the establishment of correspondence between model and image features. Furthermore, since it is not known in advance which of the models accounts for the image, the process of pose recovery is repeated separately for each of the models in the library. Consequently, methods for reducing the computational complexity of the recognition process are necessary.

An initial stage of categorization proposes a way to reduce this computational complexity. The objective of categorization is twofold. By dividing the objects into classes, a vision system is capable of concluding properties of unfamiliar objects from their resemblance to familiar ones. For familiar objects, categorization offers an indexing tool into the stored library of object representations. As an indexing tool, categorization proposes two ways to accelerate the recognition process. First, the image is compared to a smaller number of models, since only models that belong to the object's class need to be considered. Second, during categorization information about the object is extracted, and this information can be used to reduce the cost of matching the image to the individual models.

To see how information acquired during categorization can be used for identification, consider the example of face recognition. When a face is recognized, the image positions of its parts and key features are known. In particular, an observer already knows where the eyes, nose, and mouth are and can even infer the direction of

gaze and expression. The person's identity is not essential for extracting and locating these features. Instead, they are matched against features in a "generic" representation. More generally, we can postulate that, during categorization, sub-structures of the objects (such as parts and key features) are extracted and located with respect to a generic model, and the object's pose is determined.

To follow this example, I propose a scheme for recognizing 3D objects from single 2D images under orthographic projection that proceeds in two stages, categorization and identification. The library of objects is divided into classes where a class contains objects that share a fair number of similar features. (Note that such classes generally are restricted relatively to the classes considered by the human visual system.) Categorization is achieved by aligning the image to prototype objects. The prototype that appears most similar to the image determines the class identity of the object. After the object is categorized, its specific identity is determined by aligning the image to individual models of its class. By first categorizing the object not only the number of models considered for identification is reduced, but also the cost of comparing each model to the image significantly decreases. This is achieved by reusing the correspondence and pose computed for the prototype in the categorization stage to align the image with the individual models. It is shown in this paper that, albeit a perfect match between the prototype and the image is not obtainable, the correspondence and pose can be computed for the prototype, and can be used to bring the image and the object's model into alignment. Consequently, recovering the correspondence and pose for the individual models becomes unnecessary, and identification is reduced to a series of simple template comparisons.

The rest of this paper is divided as follows. Section 2 reviews the main existing approaches for categorization and identification. Section 3 presents the scheme of recognition by prototypes. Section 4 proposes an algorithm for generating optimal prototypes for the scheme. Implementation results are presented in Section 5.

2. Previous Approaches

Recognition can be performed in a variety of "abstraction levels". For example, the same object can be recognized as a face, a human face, or as a specific person's face. Psychological studies suggest the existence of a preferred level for recognition, called "the basic level of abstraction" (Rosch et al., 1976). Existing computational schemes usually approach recognition in either one of two levels. Some schemes attempt to classify objects in their basic level of abstraction (we refer to this task by *categorization*), while other schemes attempt to determine the specific identity of objects (we refer to this task by *identification*). This paper presents an attempt to combine classification with identification. However, the classes considered by our approach are generally more restricted than the basic level classes.

Existing schemes for categorization often use a "reductionist" approach. The image, which contains a detailed appearance of an object, is transformed into a compact representation that is invariant for all objects of the same class. One common approach to generating such a representation is by decomposing the object into parts. Parts are extracted by cutting the object at concavities or at inflections (Koenderink and Van Doorn, 1982; Hoffman and Richards, 1985; Vaina and Zlateva, 1990) and then labeled according to their general shape. The labels, together with the spatial relationships between the parts, are used to identify the class of the object (Binford, 1971; Marr and Nishihara, 1978; Brooks, 1981; Biederman, 1985; Bajcsy and Solina, 1987; Connell and Brady, 1987; Pentland, 1987). A second approach extracts the parts of the object that fulfill certain functions. The list of functions is used to determine the object's class (Winston et al., 1984; Ho, 1987; Stark and Bowyer, 1991; Rivlin et al., 1994).

Schemes that break objects into parts are insufficient to explain all the aspects of recognition for the following reasons. First, in many cases objects that belong to the same class differ only by their detailed shape, while they share roughly the same set of parts. Moreover, even objects that at some level may be considered belonging to different classes may also share roughly the same set of parts. To solve this problem several systems also store, in addition to the part structure of the objects, the detailed shape of the parts (Binford, 1971; Brooks, 1981; Bajcsy and Solina, 1987; Pentland, 1987). Another problem is that the existing techniques for extracting the parts from an image tend to be relatively sensitive to small changes in the image.

To recognize the specific identity of objects, a relatively detailed representation of the object's shape is compared with the image. An example for such methods is alignment (Fischler and Bolles, 1981;

Lowe, 1985; Faugeras and Hebert, 1986; Chien and Aggarwal, 1987; Thompson and Mundy, 1987; Ullman, 1989; Huttenlocher and Ullman, 1990; Basri and Ullman, 1993). Alignment involves recovering the position and orientation (*pose*) in which the object is observed and comparing the appearance of the object from that pose with the image. Only a few attempts have been made in the past to extend the alignment scheme to the problem of object categorization (e.g., Shapira and Ullman, 1991). As has already been noted, the main difficulty in applying the alignment approach is the recovery of the pose of the observed object. In most implementations this involves a time-consuming stage for finding the correspondence between the model and the image. The process becomes impractical when the image is compared against a large library of objects, because typically the correspondence is established between the image and each of the models in the library separately.

To handle large libraries, indexing methods were proposed (e.g., Lamdan et al., 1987; Weiss, 1988; Forsyth et al., 1991; Jacobs, 1992; Mundy and Zisserman, 1992; Weinshall, 1993). The basic idea is the following. A certain function is defined and applied to the views of all the objects in the library. The object models are arranged in a look-up table indexed by the obtained function values. When an image is given, the function is applied to the image, and the obtained value is used to index into the table. To reduce the size of the table and the complexity of its preparation, invariant functions, functions that when applied to different views of an object return the same value regardless of viewpoint, often are used as the indexing functions.

Indexing methods suffer from several shortcomings. First, existing indexing methods handle only rigid objects. Extending these methods to handle classes of objects has not been discussed. Second, because of complexity issues, indexing functions usually are applied to small numbers of features. As a result, high rates of false positives are obtained, and the effectiveness of the indexing is reduced.

The scheme presented in this paper differs from previous schemes in several respects. The scheme combines both categorization and identification of objects, and uses fairly detailed representations for objects. Rather than indexing directly to the specific object model, the scheme indexes into the library of objects by categorizing the object. The classes handled by the scheme include objects with relatively similar shapes. To fit into the scheme, in some cases basic level classes are broken into sub-classes. The general problem of categorization, therefore, may require additional tools.

3. Recognition by Prototypes

The recognition by prototypes scheme proceeds as follows. A library of 3D object models is stored in memory. The models in the library are divided into classes, and 3D prototype objects are selected to represent the classes. For every class, the correspondence between feature points in the prototype object and the individual models is determined, and the models are aligned in the library with the prototype. The role of this 3D alignment will become clear shortly.

At recognition time, an incoming 2D image is first matched against all of the prototypes. For each prototype object, the system attempts to recover the view of the prototype that most resembles the image. To do so, the system recovers the correspondence between the prototype and the image, and, using this correspondence, it determines the transformation that best aligns the prototype with the image. This transformation, referred to as the *prototype transform*, is then applied to the prototype, and the similarity between the transformed prototype and the actual image is evaluated. Since the observed object in general differs from the prototype object, a perfect match between the two is not anticipated. The system therefore seeks a prototype that reasonably matches the image. Once such a prototype is found, the class identity of the object is determined.

After the object's class is determined, the system attempts to recover the specific identity of the object. At this stage, the image is matched against all the models of the object's class. For each of these models, the system seeks to recover the transformation that aligns the model with the image. As will be shown below, since the models are aligned in the library with the prototype, the transformation that best aligns the prototype with the image is identical to the transformation that aligns the model to the image. The prototype transform therefore is applied to the specific models, and their appearance from this pose is compared with the image. The model that aligns with the image, if there is such, determines the specific identity of the object.

The rest of this section is divided as follows. In Section 3.1 the object representation used in our scheme is presented. Section 3.2 describes the categorization stage, and Section 3.3 describes the identification stage.

3.1. *Object Representation*

In our scheme, an object is modeled by a matrix M of size $n \times k$, where n is the number of feature points, and k, the width of M, is related to the degrees of freedom of the object. A vector $\vec{a} \in \mathcal{R}^k$, referred to as the *transform vector*, represents the transformation applied to the object in a certain view, and the object's appearance from this view is given by

$$\vec{v} = M\vec{a}. \tag{1}$$

In the rest of this section we explain the use of this notation. The notation follows from the linear combinations scheme (Ullman and Basri, 1991), which is briefly reviewed below.

Under the linear combinations scheme an object is modeled by a small set of views, each is represented by a vector containing point positions, where the points in these views are ordered in correspondence. Novel views of the object are obtained by applying linear combinations to the stored views. Additional constraints may apply to the coefficients of these linear combinations. Computing the object pose therefore requires recovering the coefficients of the linear combination that align the model with the image and verifying that the recovered coefficients indeed satisfy the constraints. The method handles rigid objects under weak-perspective projection (namely, orthographic projection followed by a uniform scaling) and under paraperspective projection (Basri, 1995). It was extended to approximate the appearance of objects with smooth bounding surfaces (Ullman and Basri, 1991; Basri and Ullman, 1993) and to handle articulated objects (Basri, 1993). In our representation, the columns of the model matrix M contain views of the object, and the coefficients of the linear combination that align the model with the image are given by the transform vector $\vec{a}$.

For concreteness, we review the linear combinations scheme for rigid objects. Consider a 3D object O that contains n feature points (X_i, Y_i, Z_i), $1 \le i \le n$. Under weak-perspective projection, the position of the object following a rotation R, translation $\vec{t}$, and scaling s is given by

$$\begin{aligned} x_i &= sr_{11}X_i + sr_{12}Y_i + sr_{13}Z_i + st_x \\ y_i &= sr_{21}X_i + sr_{22}Y_i + sr_{23}Z_i + st_y, \end{aligned} \tag{2}$$

where r_{ij} are the components of the rotation matrix, R, t_x, t_y are the horizontal and vertical components of the translation vector, $\vec{t}$, respectively, and s is the scaling factor.

Denote by $\vec{X}, \vec{Y}, \vec{Z}, \vec{x}, \vec{y} \in \mathcal{R}^n$ vectors of X_i, Y_i, Z_i, x_i and y_i values respectively, and denote $\vec{1} = (1, \ldots, 1) \in \mathcal{R}^n$, we can rewrite Eq. (2) in a vector equation as follows:

$$\begin{aligned} \vec{x} &= a_1\vec{X} + a_2\vec{Y} + a_3\vec{Z} + a_4\vec{1} \\ \vec{y} &= b_1\vec{X} + b_2\vec{Y} + b_3\vec{Z} + b_4\vec{1}, \end{aligned} \tag{3}$$

where

$$\begin{aligned} a_1 &= sr_{11} \quad b_1 = sr_{21} \\ a_2 &= sr_{12} \quad b_2 = sr_{22} \\ a_3 &= sr_{13} \quad b_3 = sr_{23} \\ a_4 &= st_x \quad b_4 = st_y. \end{aligned}$$

Therefore,

$$\vec{x}, \vec{y} \in \text{span}\{\vec{X}, \vec{Y}, \vec{Z}, \vec{1}\}. \tag{4}$$

Different views of the object are obtained by changing the rotation, scale, and translation parameters, and these changes result in changing the coefficients in Eq. (3). We may therefore conclude that all the views of a rigid object are contained in a 4D linear space.

This property, that the views of a rigid object are contained in a 4D linear space, provides a method for constructing viewer-centered representations for the object. The idea is to use images of the object to construct a basis for this space. In general, two views provide sufficiently many vectors. Therefore, any novel view is a linear combination of two views (Ullman and Basri, 1991; Poggio, 1990). The same underlying property was used by Koenderink and van Doorn (1991) to recover the affine structure of objects from two views, and by Tomasi and Kanade (1992) to recover the 3D shape of objects from multiple views.

Not every linear combination provides a valid view of a rigid object. Following the orthonormality of the row vectors of the rotation matrix, the coefficients in Eq. (3) must satisfy the two quadratic constraints

$$\begin{aligned} a_1^2 + a_2^2 + a_3^2 &= b_1^2 + b_2^2 + b_3^2 \\ a_1b_1 + a_2b_2 + a_3b_3 &= 0. \end{aligned} \tag{5}$$

When the constraints are not satisfied, distorted (by stretch or shear) pictures of the objects are generated. In case a viewer-centered representation is used, the constraints change in accordance with the selected basis. A third view of the object can be used to recover the new constraints.

For the purpose of this paper a model for a rigid object can be constructed by building the following $n \times 4$ model matrix

$$M = (\vec{X}, \vec{Y}, \vec{Z}, \vec{1}).$$

Views of the object can be constructed as follows

$$\begin{aligned} \vec{x} &= M\vec{a} \\ \vec{y} &= M\vec{b}, \end{aligned} \tag{6}$$

where $\vec{a} = (a_1, a_2, a_3, a_4)$ and $\vec{b} = (b_1, b_2, b_3, b_4)$ are the coefficients from Eq. (3). Notice that the two linear systems can be merged into one by constructing a modified model matrix in the following way

$$\begin{pmatrix} \vec{x} \\ \vec{y} \end{pmatrix} = \begin{pmatrix} M & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \end{pmatrix}. \tag{7}$$

Similar constructions can be obtained for objects with smooth bounding surfaces and for articulated objects. The width of M, k, should then be modified according to the degrees of freedom of the modeled object. As was mentioned above, viewer-centered representations can be obtained by constructing a basis for the 4D space from images of the object. Therefore, viewer-centered models can be obtained by replacing the column vectors of M with the constructed basis.

To summarize, following the linear combinations scheme we can represent an object by a matrix M and construct views of the object by applying it to transform vectors $\vec{a}$. For rigid objects not every transform vector is valid; the components of the transform vector must satisfy the two quadratic constraints. Recognition involves recovering the transform vector $\vec{a}$ and verifying that its components satisfy the two constraints. Ignoring these constraints will result in recognizing the object even when it undergoes general 3D affine transformation. In the analysis below we largely ignore the quadratic constraints. These constraints, however, can be verified both during the categorization stage as well as during the identification stage.

3.2. *Categorization*

The recognition by prototypes scheme begins by determining the object's category. This is achieved by comparing the observed object to prototype objects, objects that are "typical exemplars" for their classes. For a given prototype, the view of the prototype that most resembles the image is recovered and compared to the actual image, and the result of this comparison determines the class identity of the object.

We begin our description of the categorization stage by defining the data structures used by the scheme. A class $C = (P, \{M_1, M_2, \ldots, M_l\})$ is a pair that includes a prototype object P and a set of object models $M_1, M_2, \ldots, M_l$. Both the prototype and the models are represented by $n \times k$ matrices, where n defines the number of feature points considered, and k is related to the degrees of freedom of the objects. For the sake of simplicity we assume here that all the objects in the class share the same number of feature points, n, and that they have similar degrees of freedom. Note that similar objects tend to have similar degrees of freedom (e.g., all of them are rigid). Both assumptions are not strict, however. The scheme can be modified to tolerate both varying number of feature points as well as different degrees of freedom. The details will be discussed later in this paper. Note that the objects can be modeled by either object-centered or viewer-centered representations. In case viewer-centered representations are used we shall assume that the models represent the objects from the same range of viewpoints. However, we shall not restrict model images across objects to be taken from the same set of viewpoints.

A class in our scheme contains objects with similar shapes. These objects share roughly the same topologies, and there exists a "natural" correspondence between them. In general we shall define the natural correspondence by matching features of the same type that are nearest to each other when the two objects are viewed from corresponding viewpoints (namely, viewpoints which minimize the difference between the volumes of the objects). Consider, for instance, the two chairs in Fig. 1. Although the shapes of these chairs are different, and some parts (e.g., the arms) appear only in one chair and not in the other, a natural correspondence between features in the two objects can be determined.

In the library of models, the natural correspondence between objects is made explicit. It is specified by the order of the row vectors of the models. Specifically, given a prototype P and object models $M_1, \ldots, M_l$, we order the rows of these models such that the first feature point of P corresponds to the first feature point of each of the models $M_1, \ldots, M_l$, and so forth. The importance of this matching between the prototype and the models will become clear in the identification stage.

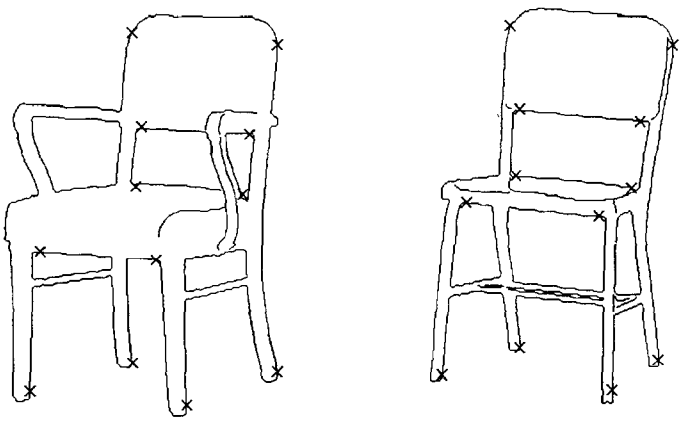

Figure 1. "Natural" correspondences between two chairs.

Given the library of objects and given an incoming image, the recognition by prototypes scheme begins by categorizing the object observed in the image. To achieve this goal, the prototype objects are aligned and compared to the image. For every prototype, the correspondence between the image and the prototype is first resolved, and, using this correspondence, the nearest prototype view is recovered. By doing so, the scheme decouples the two factors that affect the appearance of the object in the image, namely, view variations and shape variations. By selecting the nearest prototype view to the image, the scheme compensates for view variations. Then, by evaluating the similarity between the nearest prototype view and the actual image, it accounts for the differences in shape between the prototype and the observed object.

The first stage in matching the prototype to the image involves the recovery of correspondence between prototype and image features. In existing systems for recognizing the specific identity of objects establishing the correspondence between images and object models involves a time-consuming process in which sophisticated algorithms are applied (Rosenfeld et al., 1976; Davis, 1979; Fischler and Bolles, 1981; Grimson and Lozano-Pérez, 1984; Lamdan et al., 1987; Lowe, 1985; Ullman, 1989; Huttenlocher and Ullman, 1990). These algorithms rely on the property that, when the correct correspondence between a model and an image is established, a perfect match between the two is obtained. While this assumption is valid for identification, it cannot be used under our scheme since the prototype and the image generally represent different objects.

To determine the correspondence between the prototype and the image, we define an objective function that is applied to the prototype and the image under a given correspondence and that obtains its minimum under the correct correspondence (an example for such a function is given in Eq. (13)). The objective function will measure the quality of the match between the prototype and the image. Namely, under this measure the correct correspondence is the one that brings the prototype into its best alignment with the image. Given this objective function, correspondence is a combinatorial optimization problem, and so minimization techniques can be used to resolve the correspondence between the prototype and the image. In our implementation (see Section 5) we used a procedure similar to the one used in (Fischler and Bolles, 1981) to resolve the correspondence between the prototype and the image. The validity of this procedure is established in the Appendix. It is shown that when the prototype and the observed object are relatively similar the time complexity of recovering the correspondence between them using this procedure is relatively low. This procedure, however, is only one of a variety of techniques that can be used for this purpose.

After the correspondence is recovered, the scheme proceeds as follows. Given a prototype P and an image I, we generate a view vector $\vec{v}$ from the image by extracting the location of feature points and arranging them in a vector. The points in $\vec{v}$ are ordered in correspondence to the prototype points; that is, the first point in $\vec{v}$ corresponds to the first point in P and so forth. The *prototype transform* is the transformation that brings the prototype points as close as possible to their corresponding image points. The prototype transform, therefore, is the transform vector $\vec{b}$ that minimizes the Euclidean distance between the prototype and image

points, namely

$$\min_{\vec{b}'} \|P\vec{b}' - \vec{v}\|. \tag{8}$$

A solution for Eq. (8) is obtained as follows. Assuming P is overdetermined; that is, P is $n \times k$ where $n > k$ and rank$(P) = k$, and denote by $P^+ = (P^T P)^{-1} P^T$ the pseudo-inverse of P, the prototype transform, $\vec{b}$, is given by

$$\vec{b} = P^+ \vec{v}, \tag{9}$$

and the *nearest prototype view*, $\vec{p}$, is obtained by applying P to the prototype transform, $\vec{b}$, that is

$$\vec{p} = P\vec{b} = PP^+\vec{v}. \tag{10}$$

The nearest prototype view is now compared to the image, and their resemblance determines the class identity of the object. The quality of the match between the prototype and the image is defined by

$$D(P, \vec{v}) = \|\vec{p} - \vec{v}\| = \|(PP^+ - I)\vec{v}\|. \tag{11}$$

(Notice that the assumption of an overdetermined prototype is essential or else $D(P, \vec{v})$ vanish for every P and $\vec{v}$.) To eliminate effects due to scaling of the object, this measure should be normalized, as is illustrated by the example below. Consider an object seen from some view $\vec{v}_1$. Its distance to the prototype is given by $D(P, \vec{v}_1)$. Suppose the object is now seen from a new view $\vec{v}_2$ that is identical to $\vec{v}_1$, except that the object is now as twice as close to the camera. Under these conditions $\vec{v}_2 = 2\vec{v}_1$, and its distance to the prototype is given by $D(P, \vec{v}_2) = 2D(P, \vec{v}_1)$. Clearly, we should have a measure that is independent of the distance of the object to the camera. One way to obtain such a measure is by dividing $D(P, \vec{v})$ by the norm $\|\vec{v}\|$

$$\hat{D}(P, \vec{v}) = \frac{\|(PP^+ - I)\vec{v}\|}{\|\vec{v}\|}. \tag{12}$$

The normalized distance, $\hat{D}(P, \vec{v})$, has two roles. First, $\hat{D}(P, \vec{v})$ is proposed here as an objective function for establishing the correspondence between the prototype and the image. In other words, we expect that if the object belongs to the prototype's class then $\hat{D}(P, \vec{v})$ obtains its minimal value when $\vec{v}$ is ordered in correspondence to P. Any other permutation will increase the value of $\hat{D}$. Formally, denote by σ a permutation matrix, we assume that

$$\hat{D}(P, \vec{v}) = \min_{\sigma} \hat{D}(P, \sigma\vec{v}). \tag{13}$$

Secondly, since $\hat{D}(P, \vec{v})$ measures the similarity between the prototype and the image, it can also be used to determine the object's class. An object observed in a view $\vec{v}$ belongs to the class represented by a prototype P if

$$\hat{D}(P, \vec{v}) < \epsilon \tag{14}$$

for some constant $\epsilon > 0$. We refer to Eq. (14) as the *categorization criterion.*

The categorization stage proceeds as follows. Given an image I and a prototype P, the correspondence between P and I is resolved by minimizing the measure $\hat{D}(P, \sigma\vec{v})$ over all possible permutation σ of $\vec{v}$, and if the obtained minimum $\hat{D}(P, \vec{v})$ is below the threshold ϵ, then the class identity of the object is determined.

Note that in our scheme the prototype and the categorization criterion determine the actual division of objects into classes; an object belongs to a certain class if its views are sufficiently similar, according to the categorization criterion, to views of the prototype. Under the above definition, an object belongs to a prototype's class if the total (normalized) difference between its feature points and their corresponding prototype points does not exceed ϵ. Geometrically, a class is a cone of radius ϵ surrounding the column space of the prototype P.

The measure $\hat{D}(P, \vec{v})$ defined here determines the similarity between the prototype P and the view $\vec{v}$ using only the distances between feature points. In general, since correspondence is difficult to achieve, such a measure would not be robust. Including additional information about the features in the similarity measure may increase the robustness of the scheme. Also, measures that consider only the proximity of feature points are limited in terms of dividing the library into classes, since they induce classes of objects with highly similar shapes. Measures that consider additional information may extend the scheme to handle larger and more sophisticated classes of objects.

The measure $\hat{D}(P, \vec{v})$ can be enriched by considering the similarity between corresponding points. A simple example for a measure that considers both the proximity and similarity between feature points is the following measure. Each feature point is associated

with a label (such as a corner or an inflection point). Again, the measure $\hat{D}(P, \vec{v})$ is applied, but this time only correspondences between points with similar labels are allowed; namely, corners in the image can only match corners in the prototype, and, similarly, inflection points can only match inflection points. Other examples for measures that combine proximity and similarity include measures that retain the tangent or the curvature of points. More sophisticated measures may compare the topologies of the objects in the two views, or, in other words, verify that the objects share similar part structures in 2D.

A useful technique in measuring the similarity between the image and the nearest prototype view is to consider a larger set of features than the set used to determine the prototype transform. The rationale behind this technique is that it is generally difficult to recover exact feature-to-feature correspondence, and while such correspondences are necessary for recovering the prototype transform, similarity measures can be successfully applied even in the absence of exact feature-to-feature correspondence. This idea resembles the basic principle of the alignment algorithm (Ullman, 1989; Huttenlocher and Ullman, 1990), in which a small set of points is used to compute the object pose, while a larger set of points is used to verify this pose.

It should be noted that the general flow of the scheme and, in particular, the identification stage are independent of the specific choice of similarity measure. As has been noted above, the measure affects the division of model libraries into classes and the selection of optimal prototypes for these classes. An example for selecting the optimal prototype for a given class under the measure specified in Eq. (12) (for either labeled or unlabeled features) is described in Section 4.

Another important issue is the choice of threshold value, ϵ. In general, this value will depend on the structure of the classes considered by the system and on the specific similarity measure used. In particular, a different threshold value may be assigned to each of the classes. Methods for estimating the optimal threshold value for given classes of objects (such as MAP estimators) are not discussed in this paper.

Finally, although the main objective of the categorization stage is to determine the class identity of the object, the categorization scheme described above is useful even if the object's category cannot be determined. Section 3.3 below shows that the prototype transform can be reused to align the image with the specific models. Consequently, following the categorization stage the cost of comparing the image to each of the specific models is substantially reduced since the difficult part of recovering the transformation that relates the models to the image is applied only to the prototype objects. As a result, if the class identity of the object cannot be determined we still need to consider all the specific models in the library, but the overall cost of comparing the models to the image would be low because correspondence is computed once for the whole class.

3.3. *Identification*

After the observed object is categorized, the system turns to recovering its individual identity. At this stage the image is matched to all the models in the object's class. For each model, the system seeks to recover the transformation that aligns the model to the image, if there is such. In previous schemes this required recovering the correspondence between the image and each of the models separately. In our scheme, however, this no longer is necessary, since these correspondences can be inferred from the initial match between the prototype and the individual models. Thus, the model transform can be recovered directly from the prototype transform. We show in this section that the prototype and the model transforms are related by a simple transformation, which can be computed in advance, and which can in fact be undone already in the library of stored models. Consequently, the prototype transform can be reused in the identification stage to align the individual models with the image.

The initial stage of categorization recovers three pieces of information that can be used for identification. The three are: (i) the object class, (ii) the correspondence between the prototype and the image, and (iii) the prototype transform. This information is used in the identification stage as follows. First, since the object's class is determined, only models that belong to this class are considered. Second, using the correspondence between the prototype and the image established in the categorization stage, and using the stored correspondence between the prototype and each of the object models in the class, the correspondence between the models and the image is immediately recovered. Finally, as is shown below, the model transform, namely, the transformation that aligns the model with the image, is recovered from the prototype transform. This recovery is possible because the feature

points in the prototype and the models of a class are matched in advance.

Assume we are given with a view $\vec{v}$ of some object model M_i, namely

$$\vec{v} = M_i \vec{a} \tag{15}$$

for some transform vector $\vec{a}$. When the identification process begins, it is still unknown which of the models $M_1, \ldots, M_l$ of the object's class accounts for the image and what the transform vector $\vec{a}$ is. The first task faced by the scheme at this stage is to recover the model transform, $\vec{a}$. This is done, as is explained below, using the prototype transform $\vec{b} = P^+\vec{v}$ defined in Eq. (9). Once $\vec{a}$ is recovered, it is applied to all the models $M_1, \ldots, M_l$, and the model for which a near-perfect match is obtained determines the object's identity.

Theorem 1 below establishes that the model transform $\vec{a}$ can be recovered directly from the prototype transform $\vec{b}$ by applying a linear transformation which is referred to as the *prototype-to-model transform*. This transform has two interesting properties. First, it is view-independent; namely, for any given view of the object, the same transform maps the prototype transform that corresponds to this view to the correct model transform. The prototype-to-model transform therefore can be computed in advance and stored in the library of models. Second, the prototype-to-model transform can be used to recover the model transform regardless of the quality of the match between the prototype and the image. In other words, even if the prototype aligns poorly with the image, the transformation that aligns the model with the image is determined correctly in this process.

Theorem 1. *Let $\vec{v} = M_i\vec{a}$ be a view of M_i, and let $\vec{b} = P^+\vec{v}$ be the prototype transform, that is, the transform vector that best aligns the prototype with the image. If $\det(P^+M_i) \neq 0$, then the model transform, $\vec{a}$, can be recovered from the prototype transform, $\vec{b}$, by applying a matrix A_i, namely*

$$\vec{a} = A_i\vec{b}.$$

A_i is referred to as the prototype-to-model transform.

Proof: Notice that

$$\vec{b} = P^+\vec{v} = P^+M_i\vec{a}.$$

Since $\det(P^+M_i) \neq 0$ then P^+M_i is invertible. Let

$$A_i = (P^+M_i)^{-1},$$

we obtain that

$$\vec{a} = A_i\vec{b}. \qquad \square$$

Corollary 2. *The prototype-to-model transform is view-independent.*

Proof: The prototype-to-model transform, A_i, is independent of both pose vectors, $\vec{a}$ and $\vec{b}$. Changing the image $\vec{v}$ will result in a new pair of pose vectors, $\vec{a}$ and $\vec{b}$, but similar to the old pair, the new pair is related through the same transform A_i. The prototype-to-model transform A_i therefore can be used to recover the object pose for any view of M_i. $\square$

A_i exists if P^+M_i is invertible. This condition is equivalent to requiring that the two column spaces of P and M_i will not be orthogonal in any direction. The condition holds, in general, when the two objects are fairly similar. This is illustrated by the following example. Consider the case that both column spaces of P and M_i are unidimensional; namely, each represents a line through the origin. The only case in this example in which A_i does not exist is when P and M_i are orthogonal. But these lines are farthest apart when they are orthogonal. Consequently, if the objects are relatively similar A_i would exist.

Since it depends only on the prototype P and the model M_i, the prototype-to-model transform A_i can be pre-computed and stored in the library of models. Every model $M_i \in \mathcal{C}$ is associated with its own transform A_i that relates, for every possible view of M_i, between the prototype transform and the model transform. To compare the image to the model M_i the model transform should first be recovered. This is achieved by applying A_i to the prototype transform computed in the categorization stage.

Furthermore, the prototype-to-model transform, A_i, can be used to align the model M_i with the prototype P in 3D. Denote the aligned model by M_i', M_i' models the same object as M_i does, since their column vectors span the same space. In addition, the aligned model M_i' has the property that it is brought by the prototype transform, $\vec{b}$, to a perfect alignment with the image. Consequently, if the models are aligned in the library with the prototype, the prototype transform computed in the categorization stage can be reused for

identification with no further manipulations. This is established in Theorem 3 below.

Theorem 3. *Let $M_i' = M_i A_i$ be the model M_i aligned with the prototype P. For any view $\vec{v} = M_i\vec{a}$, the prototype transform for this view $\vec{b} = P^+\vec{v}$ is identical to the model transform for this view; that is, $\vec{v} = M_i'\vec{b}$.*

Proof: Since

$$M_i' = M_i A_i$$

we obtain that

$$M_i'\vec{b} = M_i A_i \vec{b} = M_i\vec{a} = \vec{v}. \qquad \square$$

Using Theorem 3, the identification scheme is simplified as follows. The models $M_1, \ldots, M_l$ are aligned in the library with the prototype P by applying the corresponding prototype-to-model transform, $A_1, \ldots, A_l$. At recognition time, the prototype transform $\vec{b} = P^+\vec{v}$, is applied to the aligned models $M_1', \ldots, M_l'$. According to Theorems 1 and 3, by transforming the models by $\vec{b}$ the correct model, M_i', would perfectly align with the image. Notice that when viewer-centered representations are used the prototype and the models image stored in the library are not required to be taken from the same set of viewpoints, since by applying the prototype-to-model transform these images will be automatically aligned.

In the scheme above we assumed that full feature-to-feature correspondence is established between the prototype and the image. This assumption is not mandatory. Methods for estimating the prototype transform using partial correspondence (e.g., under partial occlusion) or by considering other types of features (such as line segments) can also be used. Note that in case the prototype transform can only be approximated, the accuracy of the model transform obtained is determined by the quality of this approximation as well as by the condition number of the prototype-to-model transform A_i. The condition number of A_i affects the match even if Theorem 3 is applied, namely, even if the models are aligned with the prototype in advance.

The condition number of the prototype-to-model transform A_i may also be used as a criterion to divide the library into classes since it reflects the similarity between objects. For instance, a class may include all the objects M_i for which the condition number of the corresponding prototype-to-model transform A_i does not exceed some threshold ρ. Dividing the library this way guarantees that errors in estimating the prototype transform would not be amplified to corrupt the match between the specific model and the image by more than a constant factor ρ.

3.4. *Summary*

We presented in this section a scheme for recognizing 3D objects from single 2D views under orthographic projection that proceeds in two stages, categorization and identification. The main steps in this scheme are summarized below.

Preparations. In an offline stage the library of objects is divided into classes, where a class contains objects that share a fair number of similar features. A prototype object is then selected from every class (a method for constructing optimal prototypes is described in Section 4), and the feature points of the prototype are matched against the feature points of the individual models. Finally, the prototype-to-model transform is determined and applied to each of the models so as to align it (in 3D) with the prototype.

Categorization. In the categorization stage the image is compared against all the stored prototypes. For every prototype, the correspondence between the image and the prototype is recovered, and the nearest view of the prototype is constructed. The similarity between this view and the image is evaluated, and, if the two are found similar, the class identity of the object is determined. If at this stage more than one prototype is found similar to the image the information is passed on to the identification stage, and the identification procedure will be repeated for each of the respective classes.

Identification. In the identification stage the observed object is compared against all the models of its class. Since the prototype and the models were brought in the library into alignment, the same transformation that aligns the prototype to the image also aligns the object model to the image. (Alternatively, if the models were not aligned with the prototype in advance for every model in the class the prototype-to-model transform is applied to the prototype transform to obtain the transformation which relates the correct model with the image.) The prototype transform therefore is applied to the models, and the obtained views are compared with the image. The view that is found to be identical up to

noise and occlusion to the image determines the individual identity of the object.

The presented scheme is based on several key principles. Recognition is divided into two sub-processes, categorization and identification. In both stages models are aligned with the image, and the identity of the object is determined by a 2D comparison; 3D reconstruction of the observed object from the image is not performed. The difficult component of the alignment approach, namely, the recovery of correspondence and object pose, is performed only once for each class; the prototype transform is reused in the identification stage to align the image with the individual models.

4. Constructing Optimal Prototypes

In the scheme above we assumed that the classes in the library of models are represented by prototype objects. Since categorization is achieved by matching the image to prototype objects, better results can be anticipated if we select the prototype that produces the best matches. In this section we present an algorithm for constructing optimal prototypes.

Given a class of objects, the optimal prototype for this class is the object that resembles the objects of the class the most. Under our formulation, such an object would share as many features as possible with the objects of its class, the position of these features on the prototype would be as close as possible to their position on the objects, and the prototype-to-model transform for these objects would be as stable as possible. Below we show that the optimal prototype can effectively be computed using a principal components analysis; that is, by computing the eigenvectors that correspond to the dominant eigenvalue for some matrix determined by the models of the class.

Principal components analysis often is used in classification problems to reduce the dimensionality of the data while preserving the most of its variance (Duda and Hart, 1973). In existing applications, objects are represented by points in some high dimensional space, where the components of the points represent the invariant attributes of the objects. Using the principal components, the objects are mapped to a lower-dimensional space, where it is assumed that objects that belong to the same class will tend to cluster together. Alternatively, the principal components are used directly to construct classes and prototypes. In this case it is assumed that the objects that belong to the same class lie along some hyperplane in the space of all objects. The goal of the principal components analysis is, given a set of points (objects), to recover the hyperplane (class) that these points induce. Our case is somewhat different. In our case an object is represented by a continuous linear space (representing all its possible views) rather than by a point. Whereas the use of hyperplanes in other schemes often is arbitrary and made primarily for convenience, their use in our scheme is appropriate following the linear combinations scheme (Ullman and Basri, 1991; see Section 3.1).

The differences outlined above also imply differences in the proof that a principle components analysis applies to our case. We show below that the optimal prototype can be computed by principal components analysis. The traditional proof needs to be extended since in our case objects are represented by continuous spaces rather than by discrete points.

The prototype constructed in this process is a 3D object obtained by manipulating the objects in its class. To allow the construction, it seems as if the objects in the class should first be brought into alignment. In particular, if the objects are represented by viewer-centered models (that is, by sets of their views, see Section 3.1 for details), the different objects would then have to be represented by images taken from similar viewpoints. Nevertheless, the process presented below does not require an initial alignment of the objects. The same prototype is obtained in this process even when the objects are not aligned.

We now turn to constructing the optimal prototype. First, we define an objective function. Given a prototype P and an object model M_i, we define the similarity between P and M_i as follows. Let $\vec{v}_i$ be a view of M_i, we measure the similarity between the prototype P and the view $\vec{v}_i$ using Eq. (12). Then, we sum the measure over all possible views of M_i. Assuming without the loss of generality that $\|\vec{v}_i\| = 1$, Eq. (12) can be rewritten as

$$\hat{D}(P, \vec{v}_i) = \|(PP^+ - I)\vec{v}_i\|. \tag{16}$$

Without the loss of generality, we can assume that the constructed prototype, P, is composed of orthonormal columns. Note that an overdetermined matrix P with orthonormal columns satisfies $P^+ = P^T$. We can therefore rewrite Eq. (16) as

$$\hat{D}(P, \vec{v}_i) = \|(PP^T - I)\vec{v}_i\|. \tag{17}$$

The distance between P and the model M_i is now given by summing $\hat{D}(P, \vec{v}_i)$ over all unit-length (to eliminate

scaling effects) views of M_i, namely

$$\hat{D}(P, M_i) = \int_{\|\vec{v}_i\|=1} \|(PP^T - I)\vec{v}_i\| \, d\vec{v}_i. \quad (18)$$

To obtain the objective function, we sum these distances over all models

$$E(P) = \sum_{i=1}^{l} \int_{\|\vec{v}_i\|=1} \|(PP^T - I)\vec{v}_i\| \, d\vec{v}_i. \quad (19)$$

The object P that minimizes this function is defined to be the optimal prototype.

Note that Eq. (19) is not the only possible objective function for this purpose. An alternative "worst case" approach is to measure the distance between the prototype to the farthest model in the class (rather than summing this distance over all models). Except for being difficult to compute, this measure also is sensitive to "outlier" models.

The prototype that minimizes Eq. (19) can be constructed in a process that includes the following steps.

1. Verify that the column vectors of each of the model matrices, M_i $(1 \leq i \leq l)$, are orthonormal. In case they are not, apply a Gram-Schmidt process to them. (Such a process obviously does not alter the space of views implied by the models.)
2. Build the $n \times n$ symmetric matrix

$$F = \sum_{i=1}^{l} M_i M_i^T.$$

3. Find the k eigenvectors of F that correspond to its dominant eigenvalues. The optimal matrix P is constructed from these eigenvectors.

Note that, in general, we are trying to construct a prototype object that would belong to the given class. This condition determines the choice of width k for the prototype. If all the models share the same width then the prototype would assume this width. In the rigid case, for example, $k = 4$ (see Section 3.1). In case more than k large eigenvalues are obtained, one may ignore these guideline rules and construct a prototype that has higher degrees of freedom than the objects in the class. One should note, however, that the larger the rank of the prototype is the larger becomes the number of objects that can align well with the prototype. Thus, increasing the rank of the prototype will effectively increase the size of the class represented by the prototype.

Theorem 4 below establishes that the algorithm above produces the optimal prototype. We consider here the case that all the objects share similar degrees of freedom. The same procedure can be applied with slight modifications to include the case of objects with different degrees of freedom.

Theorem 4. *Let $M_1, M_2, \ldots, M_l$ be a set of models belonging to some class C. Assume every model M_i is represented by an $n \times k$ matrix with orthonormal column vectors. The prototype P that minimizes the term*

$$E(P) = \sum_{i=1}^{l} \int_{\|\vec{v}_i\|=1} \|(PP^T - I)\vec{v}_i\| \, d\vec{v}_i,$$

where the integration is performed over all the unit-length views $\vec{v}_i$ of each model M_i, is composed of the k eigenvectors of the matrix

$$F = \sum_{i=1}^{l} M_i M_i^T$$

that correspond to its k largest eigenvalues.

Proof: Let P be composed of the k eigenvectors of F that correspond to its k dominant eigenvalues. By regression principles (*best eigenvector fit*, see, e.g., Duda and Hart, 1973; p. 332) P minimizes the term

$$\sum_{i=1}^{l} \sum_{j=1}^{k} \|(PP^T - I)\vec{m}_{ij}\|,$$

where $\vec{m}_{ij}$ is the jth column vector of M_i. In other words, consider $\vec{m}_{ij}$ as a point in $\mathcal{R}^n$. The space spanned by the column vectors of P is the nearest k-dimensional hyperplane to these points, $\vec{m}_{ij}$. The rest of this proof extends the claim from the discrete sum over the column vectors of M_i to the continuous integral over all views spanned by these vectors. According to our assumptions, each matrix M_i contains an orthonormal set of column vectors. Replacing these vectors by another orthonormal basis for M_i will not change the matrix P; that is, P is independent of the choice of orthonormal basis for the models. This is illustrated by the following derivation. To obtain a new

orthonormal basis for the column space of M_i we can apply a $k \times k$ rotation matrix R to M_i (namely, $M_i R$). P is the best vector space for the new set as well, since

$$M_i R(M_i R)^T = M_i R R^T M_i^T = M_i I M_i^T = M_i M_i^T.$$

F therefore is constant for any choice of orthonormal vectors for $M_1, \ldots, M_n$, and so its eigenvectors that correspond to its dominant eigenvalues represent the best vector space for any orthonormal representation of the objects. Consequently, P minimizes the objective function regardless of choice of basis for the models, and therefore it also minimizes the required term

$$E(P) = \sum_{i=1}^{l} \int_{\|\vec{v}_i\|=1} \|(PP^T - I)\vec{v}_i\| \, d\vec{v}_i. \qquad \square$$

To summarize, we showed that given a class of object models, the optimal prototype for this class is given by the eigenvectors of the matrix F that correspond to its dominant eigenvalues, where F is constructed from the object models. Note that in proving Theorem 4 we showed that the prototype is independent of choice of basis for the models. This implies that, in order to construct the prototype, the object models $M_1, \ldots, M_l$ do not need to first be brought into alignment. The process above guarantees to output the same prototype object even if the models are not aligned in advance. An illustrative example for constructing an optimal prototype constructed using this procedure is given in Section 5.

5. Implementation

To test the ideas presented in the paper, we have implemented the scheme and applied it to several objects. In our implementation, the library of models included two classes. The first (Fig. 2) contained two four-legged chairs (denoted by A and B), and the second (Fig. 3) included two car models, a VW and a Saab. As we have used a simple categorization criterion (Eq. (14)) our program was applied to objects that belong to relatively distinct classes. Further experimentation with more sophisticated similarity measures is needed in order to distinguish between other, more similar classes.

To demonstrate categorization, we used chair A as a prototype and matched it to an image of chair B. High curvature points (such as the ones marked in Fig. 1) were selected from both the image and the prototype. Correspondences between image points and prototype points were determined by applying a procedure similar to the one proposed in Fischler and Bolles (1981). First, quadruples of image points were matched to quadruples of prototype points providing an initial estimate for the prototype transform. Then, the estimated transform was used to extend the correspondence set. Finally, the prototype transform was recomputed using

Figure 2. Pictures of two chairs used as models. We refer to these chairs by A (left) and B (right). Models for the two chairs were constructed from single images using symmetry (Poggio and Vetter, 1992).

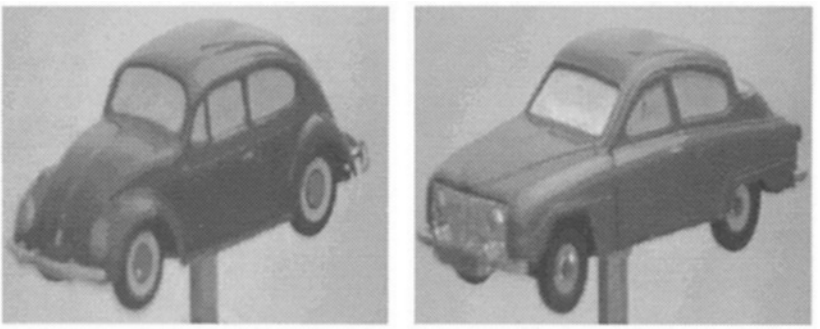

Figure 3. Pictures of two cars used as models. Left: A VW model. Right: A Saab model. Models for the two cars were borrowed from (Ullman and Basri, 1991).

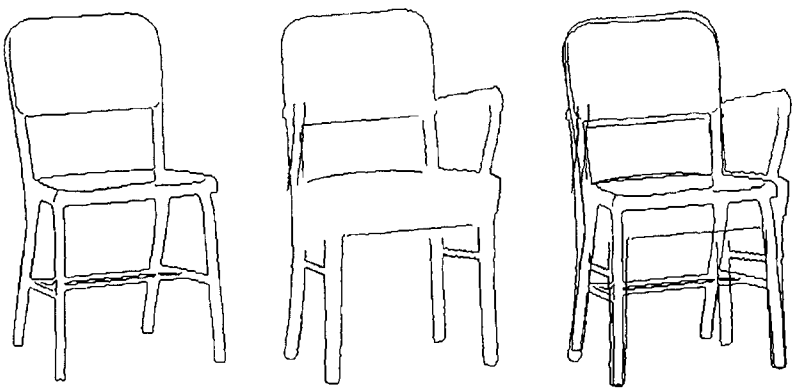

Figure 4. Matching a prototype chair (chair A) to an image of chair B. This figure, as well as the rest of the figures, contain three pictures. Left: The image to be recognized. Middle: The appearance of the prototype following the application of the prototype transform. Right: An overlay of the left and the middle pictures.

the extended correspondence set. (See discussion in the Appendix.)

The results of matching the transformed prototype with the image are seen in Fig. 4. It can be seen that the transformed prototype (middle figure) assumed the same orientation as the observed object (left figure), and that the match between the two is good considering that the objects have different shapes. Note that in this implementation we allowed the objects to undergo general affine transformations in 3D, including stretch and shear, and so the match between the prototype and the image was better than if only rigid transformations were allowed. Additional examples using chair B and the two cars as the prototypes are shown in Figs. 5–7.

In Figs. 8–9 we match the prototypes to the images while using wrong correspondences. The results of these matches are significantly worse than when the correct matches are used. This is consistent with the idea discussed in Section 3.2 that the quality of the match can be used as the objective function for resolving the correct correspondence.

Figure 10 shows the results of matching a prototype four-legged chair to a single-legged office chair. As is expected, the overall match is not very good. However, the upper portions of the chairs match relatively well, while the legs of the chairs do not find appropriate matches. This example demonstrates that evaluating a match according to distances between feature points and lines is insufficient to achieve full basic-level categorization. Evaluation procedures that examine the overall shape and topology of the compared objects may potentially improve the performance of the system.

Figure 11 shows the result of matching a prototype chair to an image of a Saab car. Anecdotally, the hole below the back of the chair was matched to the windshield of the car and the seat was matched to the hood. In general, regardless of which correspondence is used,

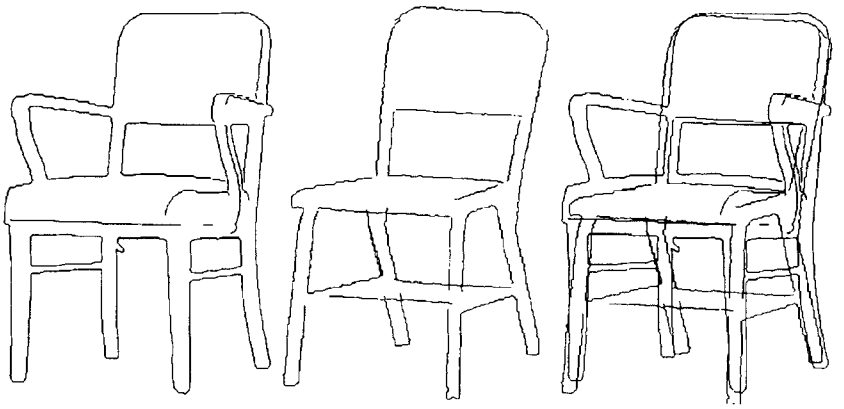

Figure 5. Matching a prototype chair (chair B) to an image of chair A.

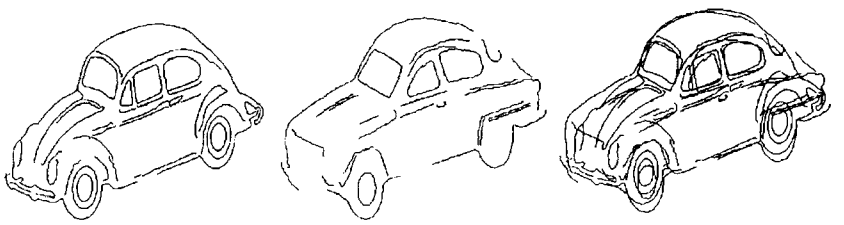

Figure 6. Matching a prototype car (Saab) to an image of a VW car.

Figure 7. Matching a prototype car (VW) to an image of a Saab car.

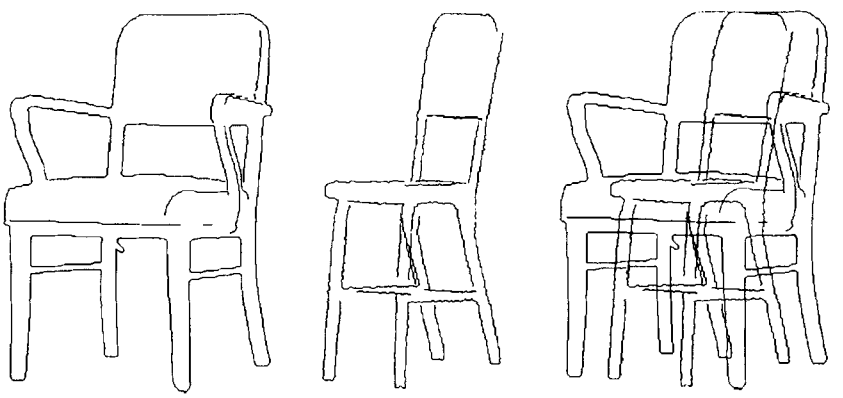

Figure 8. Matching a prototype chair (chair B) to an image of chair A with wrong correspondence.

Figure 9. Matching a prototype car (Saab) to an image of a VW car with wrong correspondence.

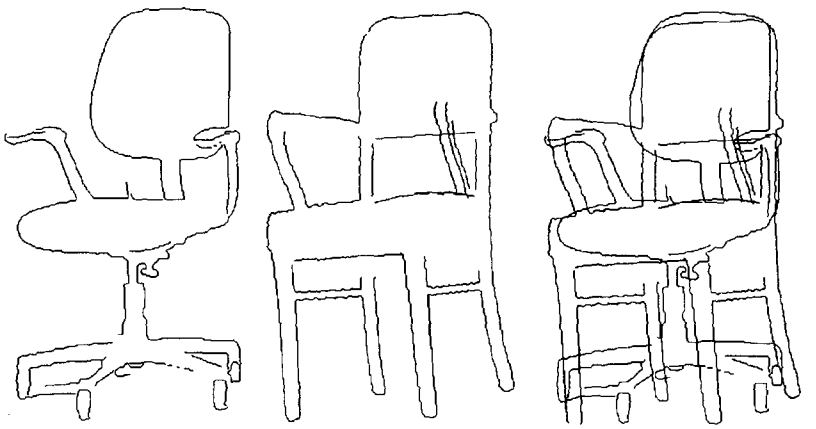

Figure 10. Matching a four-legged chair to an image of an office chair.

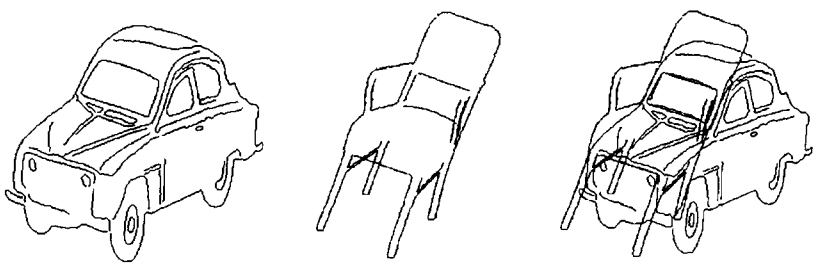

Figure 11. Matching a prototype to a chair (chair A) to an image of a Saab car.

the two objects would match poorly relative to matching the prototypes to objects of their class.

Figures 12 and 13 demonstrate the identification stage. In the library we first aligned the model for chair A with the prototype chair (chair B) using the prototype-to-model transform. Then, an image of chair A was categorized (Fig. 5) by matching it to the prototype chair, and the prototype transform was computed. In the next step, the prototype transform was applied to the specific model of chair A. The result of this application is seen in Fig. 12. It can be seen that a near-perfect alignment was achieved in this process. A similar process was applied to the VW car in Fig. 13 using the Saab car as the prototype. (The result of the corresponding categorization stage is shown in Fig. 6.) These figures demonstrate that although a perfect match between the prototype and the image could not be obtained, the prototype transform can still be used to align the observed object with its specific model.

Finally, an illustrative example demonstrating the process of constructing optimal prototypes is presented in Fig. 14. Two planar artificially-drawn lamp-shaped objects were used. The objects are modeled by images taken at different orientations. A prototype object was constructed using the process described in Section 4. The result prototype assumed the "averaged" shape of

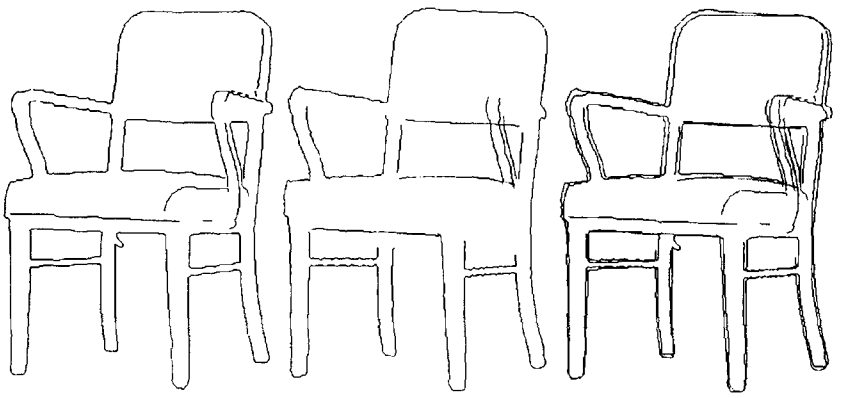

Figure 12. Matching a model of chair A to an image of the same chair using the prototype transform computed in the categorization stage.

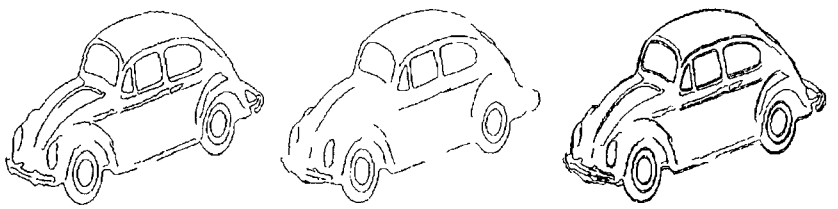

Figure 13. Matching a model of a VW car to an image of the same car using the prototype transform computed in the categorization stage.

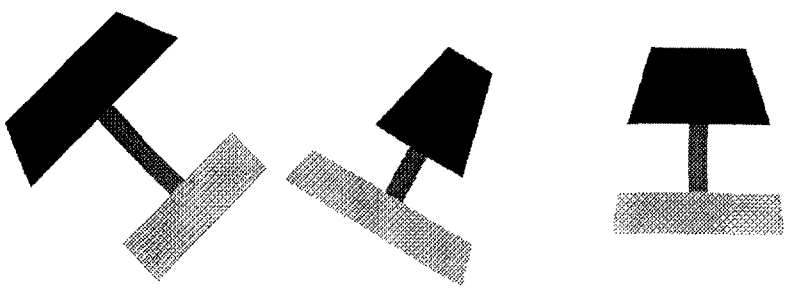

Figure 14. Constructing the optimal prototype (right) from images of two different lamp-like objects (left and middle) oriented differently.

the two original objects. Conforming with our analysis, the process was oblivious to the orientation of the object in the model images.

6. Summary

A scheme for recognizing 3D objects from single 2D images under orthographic projection was introduced. The scheme proceeds in two stages: categorization and identification. Categorization is achieved by aligning the image to prototype objects. For every prototype, the nearest prototype view is recovered, and the similarity between this view and the image is evaluated. The prototype that most resembles the observed object determines its class identity. Likewise, identification is achieved by aligning the observed object to the individual models of its class. At this stage the prototype transform computed in the categorization stage is reused to align the models with the image. The model that matches the observed object determines its specific identity. In addition, we presented an algorithm for constructing the optimal prototypes.

An important issue conveyed by our scheme is that categorization can be used to facilitate the identification of objects. We showed that by first categorizing the object, the difficult stages of the alignment process, namely, the recovery of the object pose and the

correspondence between the image and the model, can be performed only once per class. Consequently, identification is reduced in this scheme into a series of simple template comparisons.

The scheme presented in this paper differs from existing categorization schemes in two important aspects. The existing schemes (e.g., Biederman, 1985) attempt to recover the part structure (geons) of the object from the image alone. This structure is assumed to be almost invariant both to rotation of the object and across objects of the same class. In contrast, our scheme does not attempt to recover any 3D information from the image alone. Moreover, it separates the two effects that determine the object's appearance: view variation effects and deformations due to class variability. View variations are compensated for by recovering the view of the prototype that most resembles the image, and the amount of deformation that separates the prototype from the specific object is evaluated by assessing the difference (in 2D) between the nearest prototype view and the image. Note that recovering the part structure of the object from the image may be useful also in our scheme since it can be used to guide the process of establishing correspondence between the image and the prototype.

One of the limitations of our method is that it relies on matching the feature points of a prototype with points on the individual models of its class. By this we limit the scope of a class to include objects which share a fair number of feature points at proximate positions. Additional research is required in order to construct recognition schemes that combine categorization with identification while using more complicated features (such as line segments, curve pieces, and regions). Such schemes will potentially extend the scope of our method and increase its robustness.

Open problems for future research include developing efficient method for recovering the correspondence between prototypes and images, combining the scheme with existing indexing approaches (e.g., to allow direct indexing to the relevant prototype), defining effective measures to evaluate the quality of matches, handling partial occlusion, automatization of the process of dividing objects into classes, and extending the system to incorporate additional cues, such as color and texture.

Appendix

The categorization stage involves the recovery of the prototype view that resembles the image the most. A fundamental difficulty in computing this view is the need to recover the full correspondence between the feature points in the prototype and in the image. Unfortunately, enumerating all the possible correspondences is impractical since the number of possible correspondences is exponential in the number of feature points. To reduce this complexity we have implemented a matching procedure similar to the one proposed by Fischler and Bolles (1981). Below we analyze the validity of this procedure and show that, when the prototype and the observed object are sufficiently similar, this procedure can be used to recover the correspondence between the prototype and the image.

The procedure for establishing the correspondence between the prototype and the image is the following. Given a prototype P (assume P is $n \times k$) and an image I,

1. Arbitrarily select and match a subset of k prototype features to a subset of k image features. (We refer to these matches as *key correspondences*.)
2. Compute the transformation that aligns the key prototype features with their corresponding image features. Apply this transformation to the prototype.
3. Compare the transformed prototype to the image and extend the set of correspondences by adding pairs of features that fall close to each other. The transformation can be re-estimated during this process so as to improve the match between the prototype and the image.
4. Compute the transformation that best aligns the extended set of prototype features with their corresponding image features. Apply the new transformation to the prototype.
5. Evaluate the match between the transformed prototype and the image. If the match exceeds a predetermined threshold (hence satisfying Eq. (14)) the object is categorized. Otherwise, select a new set of key correspondences and repeat Steps 2–5.

Clearly, the complexity of this algorithm is polynomial in the number of feature points since we enumerate all the k-tuples of correspondences where k is independent of the number of features.

There is one issue to be resolved, however. The Fischler and Bolles's algorithm was developed to solve an identification task, that is, to compare a model to an image of the same object. The algorithm, therefore, is built around the premise that the transformation determined by any subset of key correspondences is identical to the transformation that aligns the entire model with

the image. In addition, the algorithm assumes that when this transformation is applied to the model the rest of the object's features will coincide (or fall near in case of reasonable errors) with their corresponding image features. Our case is different, however. In our case the prototype and the image may represent different objects. Thus, different choices of correct partial correspondence will produce transformations which differ from the sought prototype transform, and the transformed prototype features cannot be expected to coincide with their corresponding image features. Since in our algorithm features are matched according to a proximity criterion (Step 3) we have to verify that, at least for some choices of partial correspondence, corresponding features will indeed fall close to each other. In the rest of this appendix we analyze the effect of using partial correspondence on estimating the prototype transform. Bounds on the position of the transformed prototype features are computed, and the conditions for obtaining good estimates of the prototype transform are derived.

We follow Jacobs (1992) in this analysis. Jacobs analyzed the effect of errors on the predicted position of image features in an identification task of planar objects. He found that the deviations between the predicted and actual positions of the points are the result of the errors in the key points amplified by the affine coordinates of the rest of the points with respect to the key points. Below we extend his analysis to the case of comparing a prototype to an object. Here the "errors" represent differences in shape between the prototype and the observed object. The analysis is extended also to objects of arbitrary dimension.

Given an $n \times k$ prototype P and a corresponding image vector $\vec{v} \in \mathcal{R}^n$, the sought prototype transform, $\vec{b}$, defined in Eq. (9), is given by

$$\vec{b} = P^+ \vec{v} \tag{20}$$

and the corresponding prototype view, $\vec{p}$, by

$$\vec{p} = P\vec{b}. \tag{21}$$

In our algorithm we initially estimate the prototype transform by matching k prototype and image features. Assume without the loss of generality that the k matched features are ordered in the top of P and $\vec{v}$. Denote the top $k \times k$ sub-matrix of P by Q (assuming Q is non-singular) and the top k components of $\vec{v}$ and $\vec{p}$ by $\vec{u}$ and $\vec{q}$ respectively ($\vec{u}, \vec{q} \in \mathcal{R}^k$). The prototype transform is estimated by

$$\vec{c} = Q^{-1}\vec{u}, \tag{22}$$

and the prototype view corresponding to this estimated transform is

$$\vec{p}\,' = P\vec{c} = PQ^{-1}\vec{u}. \tag{23}$$

In the next step, we attempt to extend the set of correspondences. At this stage it is necessary that the difference between the predicted prototype view and the actual image, $\|\vec{p}\,' - \vec{v}\|$, will be small. Below we derive a bound on $\|\vec{p}\,' - \vec{v}\|$. To derive the bound we will express the vectors $\vec{p}$ and $\vec{p}\,'$ with respect to their first k components. The first k components of the best prototype view, $\vec{p}$, were denoted by $\vec{q} \in \mathcal{R}^k$. The first k components of the estimated prototype view $\vec{p}\,'$ are identical to the first k components of the image, $\vec{v}$, and these components were denoted by $\vec{u} \in \mathcal{R}^k$. Below we show that both $\vec{p}$ and $\vec{p}\,'$ are related to their first k components ($\vec{q}$ and $\vec{u}$ respectively) by a single $n \times k$ matrix A, namely,

$$\vec{p} = A\vec{q} \tag{24}$$

and

$$\vec{p}\,' = A\vec{u}, \tag{25}$$

where A is given by

$$A = PQ^{-1}. \tag{26}$$

Equation (24) can be derived as follows. Recall that Q is the top $k \times k$ submatrix of P, and so $Q\vec{b}$ contains the top k components of $\vec{p}$, that is

$$Q\vec{b} = \vec{q}. \tag{27}$$

Consequently,

$$\vec{p} = P\vec{b} = PQ^{-1}Q\vec{b} = AQ\vec{b} = A\vec{q}. \tag{28}$$

Equation (25) follows immediately from Eq. (23).

Notice that A contains the affine coordinates of the prototype feature points. Every row l in A contains the k affine coordinates of the lth point with respect to the first k points. A therefore satisfies

$$P = AQ. \tag{29}$$

Equations (24) and (25) simply reflect the fact that affine coordinates are invariant under affine transformations. Therefore, whether we apply the prototype

transform, $\vec{b}$, to P, or whether we apply its approximated value, $\vec{c}$, to P, the affine coordinates of the lth point will remain unchanged.

We now turn to estimating the difference between the estimated prototype view and the image, $\|\vec{p}\,' - \vec{v}\|$. First, we use the triangular inequality

$$\|\vec{p}\,' - \vec{v}\| \leq \|\vec{p}\,' - \vec{p}\| + \|\vec{p} - \vec{v}\|. \tag{30}$$

Equations (24) and (25) imply that

$$\|\vec{p}\,' - \vec{p}\| = \|A(\vec{u} - \vec{q})\| \leq \|A\|\|\vec{u} - \vec{q}\|, \tag{31}$$

where $\|A\|$ denotes the max-norm of A defined by

$$\max_{x \in \mathcal{R}^k} \frac{\|Ax\|}{\|x\|}. \tag{32}$$

Since $\vec{u}$ and $\vec{q}$ contain the first k components of $\vec{v}$ and $\vec{p}$ respectively then

$$\|\vec{u} - \vec{q}\| \leq \|\vec{v} - \vec{p}\|, \tag{33}$$

and so

$$\|\vec{p}\,' - \vec{p}\| \leq \|A\|\|\vec{v} - \vec{p}\|. \tag{34}$$

Equations (30) and (34) imply that

$$\|\vec{p}\,' - \vec{v}\| \leq (\|A\| + 1)\|\vec{v} - \vec{p}\|. \tag{35}$$

According to Eq. (35) the difference between the predicted position of the prototype points and their corresponding image points is determined by two terms. One term, $\|\vec{v} - \vec{p}\|$, represents the difference between the position of feature points in the image and their corresponding points in the best prototype view. This term is small when the classes of objects are restricted to include only relatively similar objects. The other term depends on the norm of the matrix A, which contains the affine coordinates of the prototype points. This term depends on the choice of key correspondences. In particular, the norm of A will be small when the prototype points lie within or close to the convex hull of the k key correspondences.

This analysis is further emphasized if we consider the deviation in position of particular feature points. Suppose that the difference between every feature point in the best prototype view and the corresponding point in the image is bounded by some scalar, δ, namely, $|p_i - v_i| < \delta$ $(1 \leq i \leq n)$. Consider the difference in the lth point of the estimated prototype view and the image

$$|p_l' - v_l| \leq |p_l' - p_l| + |p_l - v_l|. \tag{36}$$

Now,

$$|p_l - v_l| < \delta, \tag{37}$$

and,

$$|p_l' - p_l| = \left| \sum_{i=1}^{k} a_{li}(p_i - v_i) \right| < \sum_{i=1}^{k} |a_{li}|\delta, \tag{38}$$

where a_{li} are the components of A. Equations (37) and (38) imply that

$$|p_l' - v_l| < \left(\sum_{i=1}^{k} |a_{li}| + 1 \right)\delta. \tag{39}$$

Note that if, for example, p_l lies inside the convex hull of $p_1, \ldots, p_k$ then $a_{li} \geq 0$ $(1 \leq i \leq k)$ and $\sum_{i=1}^{k} a_{li} = 1$. Consequently,

$$|p_l' - v_l| < 2\delta. \tag{40}$$

To summarize, in this appendix we have analyzed the applicability of Fischler and Bolles's algorithm to the problem of recovering the correspondence between prototype and image features. We showed that this procedure can be applied successfully to classes that contain objects of relatively similar shapes. For these objects there exist "good" choices of key correspondences, which do not amplify the deviations between corresponding features beyond a certain bound. Enumerating all the possible sets of key correspondences can in these cases guarantee the recovery of the correspondence between the prototype and the image.

Acknowledgment

I thank Shimon Ullman for encouragement and advice, Tao Alter and Yael Moses for many fruitful discussions, Dror Bar Natan for his assistance in verifying the proof for Theorem 4, Eric Grimson, John Harris, David Jacobs, and Tomaso Poggio for comments on earlier drafts. A partial version of this study was reported in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* (*CVPR-93*), New York, NY, June 1993. This report describes research done in part at the Artificial Intelligence Laboratory of the Massachusetts

Institute of Technology and the McDonnell-Pew Center for Cognitive Neuroscience. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-91-J-4038.

References

Bajcsy, R. and Solina, F., 1987. Three dimensional object representation revisited. *Proc. of The First Int. Conf. on Computer Vision*, London, pp. 231–240.

Basri, R. 1993. Viewer-centered representations in object recognition: A computational approach. In C.H. Chen, L.F. Pau, and P.S.P. Wang (Eds.), *Handbook of Pattern Recognition and Computer Vision*. World Scientific Publishing Company: Singapore, Vol. 5, No. 4, 863–882.

Basri, R. 1995. Paraperspective ≡ affine. *Int. Journal of Computer Vision*, forthcoming.

Basri, R. and Ullman, S. 1993. The alignment of objects with smooth surfaces. *CVGIP: Image Understanding*, 57(3):331–345.

Biederman, I. 1985. Human image understanding: Recent research and a theory. *Computer Vision, Graphics, and Image Processing*, 32:29–73.

Binford, T.O. 1971. Visual perception by computer. *IEEE Conf. on Systems and Control*.

Brooks, R. 1981. Symbolic reasoning among 3-dimensional models and 2-dimensional images. *Artificial Intelligence*, 17:285–349.

Chien, C.H. and Aggarwal, J.K. 1987. Shape recognition from single silhouette. *Proc. of The First Int. Conf. on Computer Vision*, London, pp. 481–490.

Connell, J.H. and Brady, M. 1987. Generating and generalizing models of visual objects. *Artificial Intelligence*, 31:159–183.

Davis, L.S. 1979. Shape matching using relaxation techniques. *IEEE Trans. on Pattern Analysis and Machine Intel.*, 1(1):60–72.

Duda, R.O. and Hart, P.E. 1973. *Pattern Classification and Scene Analysis*. Wiley-Interscience Publication, John Wiley and Sons, Inc.

Faugeras, O.D. and Hebert, M. 1986. The representation, recognition and location of 3D objects. *Int. J. Robotics Research*, 5(3):27–52.

Fischler, M.A. and Bolles, R.C. 1981. Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Com. of the A.C.M.*, 24(6):381–395.

Forsyth, D., Mundy, J.L., Zisserman, A., Coelho, C., Heller, A., and Rothwell, C. 1991. Invariant descriptors for 3-D object recognition and pose. *IEEE Trans. on Pattern Analysis and Machine Intel.*, 13:971–991.

Grimson, W.E.L. and Lozano-Pérez, T. 1984. Model-based recognition and localization from sparse data. *Int. Journal of Robotics Research*, 3:3-35.

Ho, S. 1987. Representing and using functional definitions for visual recognition. Ph.D. Dissertation, University of Wisconsin, Madison.

Hoffman, D.D. and Richards, W. 1985. Parts of recognition. *Cognition*, 18:65–96.

Huttenlocher, D.P. and Ullman, S. 1990. Recognizing solid objects by alignment with an image. *Int. Journal of Computer Vision*, 5(2):195–212.

Jacobs, D.W. 1992. Space efficient 3D model indexing. *Proc. of Image Understanding Workshop*, pp. 717–725.

Koenderink, J.J. and Van Doorn, A.J. 1982. The shape of smooth objects and the way contours end. *Perception*, 11:129–137.

Koenderink, J. and van Doorn, A. 1991. Affine structure from motion. *Journal of the Optical Society of America*, 8(2):377–385.

Lamdan, Y., Schwartz, J.T., and Wolfson, H. 1987. On recognition of 3-D objects from 2-D images. Courant Inst. of Math. Sci., Rob. TR 122.

Lowe, D.G. 1985. Three-dimensional object recognition from single two-dimensional images. Courant Inst. of Math. Sci., Rob. TR 202.

Marr, D. and Nishihara, H.K. 1978. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc. of the Royal Society, London B*, 200:269–294.

Mundy, J.L. and Zisserman, A. 1992. *Geometric Invariance in Computer Vision*. M.I.T. Press.

Pentland, A. 1987, Recognition by parts. *Proc. of the First Int. Conf. on Computer Vision*, pp. 612–620.

Poggio, T. 1990. 3D object recognition: On a result by Basri and Ullman. TR 9005-03, IRST, Povo, Italy.

Poggio, T. and Vetter, T. 1992. Recognition and structure from one 2D model view: observations on prototypes, object classes, and symmetries. M.I.T., A.I. Memo No. 1347.

Rivlin, E., Dickenson, S., and Rosenfeld, A. 1994. Recognition by Functional Parts. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 267–275.

Rosch, E., Mervis, C.B., Gray, W.D., Johnson, D.M., and Boyes-Braem, P. 1976. Basic objects in natural categories. *Cognitive Psychology*, 8:382–439.

Rosenfeld, A., Hummel, R., and Zucker, S. 1976. Scene labeling by relaxation operations. *IEEE Trans. on System and Man Cybernetics*, 7:420–433.

Shapira, Y. and Ullman, S. 1991. A pictorial approach to object classification. *Proc. of the 12th Int. Conf. on Artificial Intelligence*, pp. 1257–1263.

Stark, L. and Bowyer, K. 1991. Achieving generalized object recognition through reasoning about association of function to structure. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):992–1006.

Thompson, D.W. and Mundy J.L. 1987. Three dimensional model matching from an unconstrained viewpoint. *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 208–220.

Tomasi, C. and Kanade, T. 1992. Factoring image sequences into shape and motion. *Int. Journal of Computer Vision*, 9(2).

Ullman, S. 1989. Aligning pictorial descriptions: An approach to object recognition. *Cognition*, 32(3):193–254.

Ullman, S. and Basri, R. 1991. Recognition by linear combinations of models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):992–1006.

Vaina, L.M. and Zlateva, S.D. 1990. The largest convex patches: A boundary-based method for obtaining object parts. *Biological Cybernetics*, 62:225–236.

Weinshall, D. 1993. Model-based invariants for 3D vision. *International Journal of Computer Vision*, 10(1):27–42.

Weiss, I. 1988. Projective invariants of shape. *DARPA Image Understanding Workshop*, pp. 1125–1134.

Winston, P.H., Binford, T.O., Katz, B., and Lowry, M. 1984. Learning physical description from functional definitions, examples and precedents. M.I.T., A.I. Memo 679.