Probabilistic Models of Appearance for 3-D Object Recognition

Arthur R. Pope
David Sarnoff Research Center
apope@sarnoff.com

David G. Lowe
University of British Columbia
lowe@cs.ubc.ca

Abstract

We describe how to model the appearance of a 3-D object using multiple views, learn such a model from training images, and use the model for object recognition. The model uses probability distributions to describe the range of possible variation in the object's appearance. These distributions are organized on two levels. Large variations are handled by partitioning training images into clusters corresponding to distinctly different views of the object. Within each cluster, smaller variations are represented by distributions characterizing uncertainty in the presence, position, and measurements of various discrete features of appearance. Many types of features are used, ranging in abstraction from edge segments to perceptual groupings and regions. A matching procedure uses the feature uncertainty information to guide the search for a match between model and image. Hypothesized feature pairings are used to estimate a viewpoint transformation taking account of feature uncertainty. These methods have been implemented in an object recognition system, OLIVER*. Experiments show that* OLIVER *is capable of learning to recognize complex objects in cluttered images, while acquiring models that represent those objects using relatively few views.*

International Journal of Computer Vision, **40,** 2 (2000), pp. 149-167.

1 Introduction

Object recognition requires a model of appearance that can be matched to new images. In this paper, a new model representation will be described that can be derived automatically from a sample of images of the object. The representation models an object by a probability distribution that describes the range of possible variation in the object's appearance. Large and complex variations are handled by dividing the range of appearance into a conjunction of simpler probability distributions. This approach is general enough to model almost any range of appearance, whether arising from different views of a 3-D object or from different instances of a generic object class.

The probability distributions of individual features can help guide the matching process that underlies recognition. Features whose presence is most strongly correlated with that of the object can be given priority during matching. Features with the best localization can contribute most to an estimate of the object's position, while features whose positions vary most can be sought over the largest image neighborhoods. We hypothesize initial pairings between model and image features, use them to estimate an aligning transformation, use the transformation to evaluate and choose additional pairings, and so on, pairing as many features as possible. The transformation estimate includes an estimate of its uncertainty derived from the uncertainties of the paired model and image features. Potential feature pairings are evaluated using the transformation, its uncertainty, and topological relations among features so that the least ambiguous pairings are adopted earliest, constraining later pairings. The method is called *probabilistic alignment* to emphasize its use of uncertainty information.

Two processes are involved in learning a multiple-view model from training images (Fig. 1). First, the training images must be clustered into groups that correspond to distinct views of the object. Second, each group's members must be generalized to form a model view characterizing the most representative features of that group's images. Our method couples these two processes in such a way that clustering decisions consider how well the resulting groups can be generalized, and how well those generalizations describe the training images. The multiple-view model produced thus achieves a balance between the number of views it contains, and the descriptive accuracy of those views.

2 Related research

In recent years, there has been growing interest in modeling 3-D objects with information derived from a set of 2-D views (Breuel, 1992; Murase and Nayar, 1995). For an object that is even moderately complex, however, many qualitatively distinct views may be needed. Thus, a multiple-view representation may require considerably more space and complexity in matching than a 3-D one. Space requirements can be reduced somewhat by allowing views to share common structures (Burns and Riseman, 1992) and by merging similar views after discarding features too fine to be reliably discerned (Petitjean, Ponce, and Kriegman, 1992). In this paper, we develop a representation that combines nearby views over a wider range of appearance by representing the probability distribution of fea-

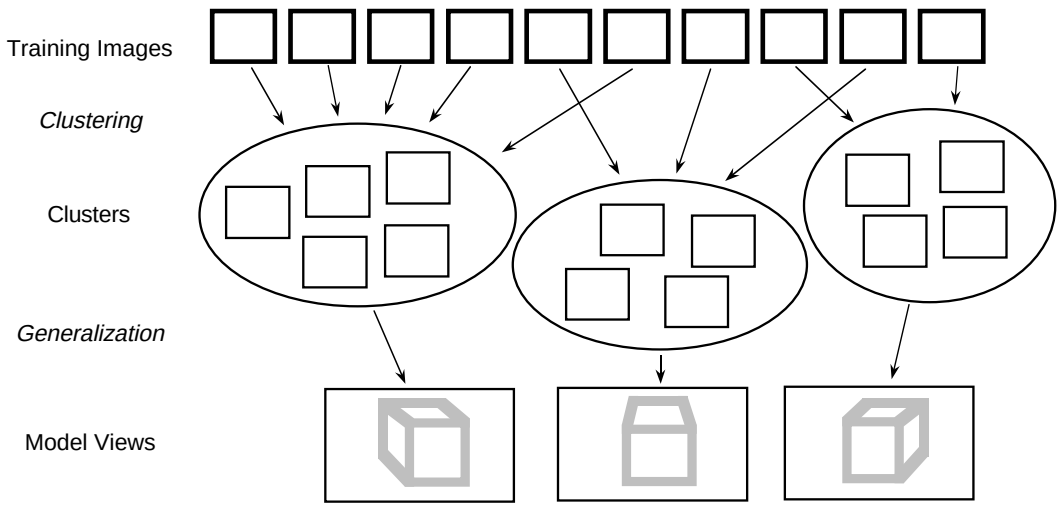


Figure 1: Learning a multiple-view model from training images requires a clustering of the training images and a generalization of each cluster's contents.

tures over a range of views.

One method for improving the space/accuracy trade-off of a multiple-view representation is to interpolate among views. Ullman and Basri (1991) have shown that with three views of a rigid object whose contours are defined by surface tangent discontinuities, one can interpolate among the three views with a linear operation to produce other views under orthographic projection. If the object has smooth contours instead, six views allow for accurate interpolation. However, for non-rigid or generic models, a more direct form of sampling and linear interpolation can be more general while giving adequate accuracy, as described in this paper.

There has also been recent development of methods using dense collections of local features, with rotational invariants computed at corner points (Schmid and Mohr, 1997). This approach has proved very successful with textured objects, but is less suited to geometrically defined shapes, particularly under differing illumination. The approach described in this paper can be extended to incorporate any type of local feature into the model representation, so a future direction for improvement would be to add local image-based features. Initially, the system has been demonstrated with edge-based features that are less sensitive to illumination change.

Other approaches to view-based recognition include color histograms (Swain and Ballard, 1991), eigenspace matching (Murase and Nayar, 1995), and receptive field histograms (Schiele and Crowley, 1996). These approaches have all been demonstrated successfully on isolated or pre-segmented images, but due to their more global features it has been difficult to extend them to cluttered and partially occluded images, particularly for objects lacking distinctive feature statistics.

2.1 Matching with uncertainty

One general strategy for object recognition hypothesizes specific viewpoint transformations and tests each hypothesis by finding feature correspondences that are consistent with it. This strategy was used in the first object recognition system (Roberts, 1965), and it has been used in many other systems since then (Brooks, 1981; Bolles and Cain, 1982; Lowe, 1985; Grimson and Lozano-Pérez, 1987; Huttenlocher and Ullman, 1990; Nelson and Selinger, 1998).

An example of this approach is the iterative matching in the SCERPO system (Lowe, 1987). A viewpoint transformation is first estimated from a small set of feature pairings. This transformation is used to predict the visibility and image location of each remaining model feature. For each of these projected model features, potential pairings with nearby image features are identified and evaluated according to their expected reliability. The best ranked pairings are adopted, all pairings are used to produce a refined estimate of the transformation, and the process is repeated until acceptable pairings have been found for as many of the model features as possible. This paper describes an enhanced version of iterative matching that incorporates feature uncertainty information.

In a related approach, Wells (1997) has shown how transformation space search can be cast as an iterative estimation problem solved by the EM algorithm. Using Bayesian theory and a Gaussian error model, he defines the posterior probability of a particular set of pairings and a transformation, given some input image. In more recent work, Burl, Weber, and Perona (1998) provide a probabilistic model giving deformable geometry for local image patches. The current paper differs from these other approaches by deriving a clustered view-based representation that accounts for more general models of appearance, incorporating different individual estimates of feature uncertainty, and making use of a broader range of features and groupings.

2.2 Use of uncertainty information in matching

Iterative alignment has been used with a Kalman filter to estimate transformations from feature pairings in both 2D-2D matching (Ayache and Faugeras, 1986) and 2D-3D matching (Hel-Or and Werman, 1995). Besides being efficient, this allows feature position uncertainty to determine transformation uncertainty, which in turn is useful in predicting feature positions in order to rate additional feature pairings (Hel-Or and Werman, 1995). However, this (partial) least-squares approach can only represent uncertainty in either image or model features, not both; total least squares can represent both, but may not be accurate in predicting feature positions from the estimated transformation (Van Huffel and Vandewalle, 1991). Most have chosen to represent image feature uncertainty; we have chosen to emphasize model feature uncertainty, which in our case carries the most useful information.

3 Model representation

An object model is organized on two levels so that it can describe the object's range of appearance both fully and accurately. At one level, large variations in appearance are handled by subdividing the entire range of variation into discrete subranges corresponding to distinctly different views of the object; this is a multiple-view representation. At a second level, within each of the independent views, smaller variations are described by probability distributions that characterize the position, attributes, and probability of detection for individual features.

The only form of appearance variation not represented by the model is that due to varying location, orientation, and scale of the object within the image plane. Two mechanisms accommodate this variation. One is the viewpoint transformation, which aligns a model view with an appropriate region of the image; we shall describe it in section 4. The other is the use of position-invariant representations for attributes, which allow feature attributes to be compared regardless of the feature positions.

3.1 Simplifying approximation of feature independence

Our method lets each model view describe a range of possible appearances by having it define a joint probability distribution over image graphs. However, because the space of image graphs is enormous, it is not practical to represent or learn this distribution in its most general form. So instead, the joint distribution is approximated by treating its component features as though they were independent. This approximation allows the joint distribution to be decomposed into a product of marginal distributions, thereby greatly simplifying the representation, matching, and learning of models.

One consequence of this simplification is that statistical dependence (association or covariance) among model features cannot be accurately represented within a single model view. Consider, for example, an object whose features are divided among two groups, only one of which appears in any instance. With its strongly covariant features, this object would be poorly represented by a single view. However, where one view cannot capture an important statistical dependence, multiple views can. In this example, two model views, each containing one of the two subsets of features, could represent perfectly the statistical dependence among them.

By using a large enough set of views, we can model any object as accurately as we wish. For economy, however, we would prefer to use relatively few views and let each represent a moderate range of possible appearances. The model learning procedure described in section 6 gives a method for balancing the competing aims of accuracy and economy.

3.2 Model view representation

A single model view is represented by a model graph. A model graph has nodes that represent features, and arcs that represent composition and abstraction relations among features. Each node records the information needed to estimate three probability distributions characterizing its feature:

1. *The probability of observing this feature in an image depicting the modeled view of the object*. This is estimated from a record of the number of times the model feature has been identified in training images by being matched to a similar image feature.

2. *Given that this feature is observed, the probability of it having a particular position.* This is characterized by a probability distribution over feature positions. We approximate this distribution as Gaussian to allow use of an efficient matching procedure based on least squares estimation. The parameters of the distribution are estimated from sample feature positions acquired from training images.

3. *Given that this feature is observed, the probability of it having particular attribute values*. This is characterized by a probability distribution over vectors of attribute values. Little can be assumed about the form of this distribution because it may depend on many factors: the type of feature, how its attributes are measured, possible deformations of the object, and various sources of measurement error. Thus we use a non-parametric density estimator that makes relatively few assumptions. To support this estimator, the model graph node records sample attribute vectors acquired from training images.

3.3 Model notation

An object's appearance is modeled by a set of model graphs $\{G_i\}$. A model graph G_i is a tuple $\langle F, R, m \rangle$, where F is a set of model features, R is a relation over elements of F, and m is the number of training images used to produce G_i.

A model feature $j \in F$ is represented by a tuple of the form $\langle t_j, m_j, A_j, B_j \rangle$. Feature j's type is represented by t_j, whose value is one of a set of symbols denoting different types of features. The element m_j specifies in how many of the m training images feature j was found. The series A_j contains the attribute vectors of those training image features that matched j. The dimension and interpretation of these vectors depend on j's type. The series B_j contains the mean positions of the training image features that matched j. These positions, although drawn from separate training images, are expressed in a single, common coordinate system, which is described in the following section.

From j's type t_j, one can determine whether j is a feature that represents a grouping or abstraction of other features. If so then R will contain a single element, $\langle k, l_1, \ldots, l_n \rangle$, specifying j's parts as being l_1 through l_n. The number of parts n may depend on j. Moreover, any l_i may be the special symbol $\perp$, which indicates that the part is not defined, and perhaps not represented in the model graph.

4 Coordinate systems and viewpoint transformations

A feature's position is specified by a 2–D location, orientation, and scale. Image features are located in an image coordinate system of pixel rows and columns. Model features are located in a model coordinate system shared by all features within a model graph.

Two different schemes are used to describe a feature's position in either coordinate system:

$xy\theta s$ The feature's location is specified by $[x\ y]$, its orientation by θ, and its scale by s.

$xyuv$ The feature's location is specified by $[x\ y]$, and its orientation and scale are represented by the direction and length of the 2–D vector $[u\ v]$.

We shall use the $xy\theta s$ scheme for measuring feature positions, and the $xyuv$ scheme to provide a linear approach for aligning features in the course of matching a model with an image. They are related by $\theta = tan^{-1}(v/u)$ and $s = \sqrt{u^2 + v^2}$. Where it is not otherwise clear we shall indicate which scheme we are using with the superscripts $^{xy\theta s}$ and ^{xyuv}.

The task of matching a model with an image includes that of determining a viewpoint transformation that closely aligns image features with model features. The viewpoint transformation, T, is a mapping from 2–D image coordinates to 2–D model coordinates—it transforms the position of an image feature to that of a model feature.

4.1 Similarity transformations

A 2–D similarity transformation can account for translation, rotation, and scaling of an object's projected image. It does not account for effects of rotation in depth, nor changes in perspective as an object moves towards or away from the camera.

A 2–D similarity transformation decomposed into a rotation by θ_t, a scaling by s_t, and a translation by $[x_t\ y_t]$, in that order, can be expressed as a linear operation using the $xyuv$ scheme, as Ayache and Faugeras (1986), among others, have done. The linear operation has two formulations in terms of matrices. We shall present both formulations here, and have occasion to use both in section 5.

We shall develop the formulations by first considering the transformation of a point location from $[x_k\ y_k]$ to $[x'_k\ y'_k]$. We can write it as

$$\begin{bmatrix} x'_k \\ y'_k \end{bmatrix} = s_t \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \end{bmatrix} \tag{1}$$

Defining $u_t = s_t \cos\theta_t$ and $v_t = s_t \sin\theta_t$ allows us to rewrite this as either

$$\begin{bmatrix} x'_k \\ y'_k \end{bmatrix} = \begin{bmatrix} u_t & -v_t \\ v_t & u_t \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \end{bmatrix} \tag{2}$$

or

$$\begin{bmatrix} x'_k \\ y'_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_k & -y_k \\ 0 & 1 & y_k & x_k \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \end{bmatrix}. \tag{3}$$

Now consider a vector $[u_k\ v_k]$ whose direction represents an orientation and whose magnitude represents a length. When mapped by the same transformation, this vector must

be rotated by θ_t and scaled by s_t to preserve its meaning. Continuing to use $u_t = s_t \cos\theta_t$ and $v_t = s_t \sin\theta_t$, we can write the transformation of $[u_k\ v_k]$ as either

$$\begin{bmatrix} u'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} u_t & -v_t \\ v_t & u_t \end{bmatrix} \begin{bmatrix} u_k \\ v_k \end{bmatrix} \tag{4}$$

or

$$\begin{bmatrix} u'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} 0 & 0 & u_k & -v_k \\ 0 & 0 & v_k & u_k \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \end{bmatrix}. \tag{5}$$

Equations 3 and 5 together give us one complete formulation of the transformation. We can write it with a matrix $\mathbf{A}_k$ representing the position $\mathbf{b}_k = [x_k y_k u_k v_k]$ being transformed, and a vector $\mathbf{t}$ representing the transformation:

$$\mathbf{b}'_k = \begin{bmatrix} x'_k \\ y'_k \\ u'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_k & -y_k \\ 0 & 1 & y_k & x_k \\ 0 & 0 & u_k & -v_k \\ 0 & 0 & v_k & u_k \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \end{bmatrix} = \mathbf{A}_k \mathbf{t}. \tag{6}$$

Equations 2 and 4 together give us another complete formulation. We can write it with a matrix $\mathbf{A}_t$ representing the rotation and scaling components of the transformation, and a vector $\mathbf{x}_t$ representing the translation components:

$$\mathbf{b}'_k = \begin{bmatrix} x'_k \\ y'_k \\ u'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} u_t & -v_t & 0 & 0 \\ v_t & u_t & 0 & 0 \\ 0 & 0 & u_t & -v_t \\ 0 & 0 & v_t & u_t \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ u_k \\ v_k \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ 0 \\ 0 \end{bmatrix} = \mathbf{A}_t \mathbf{b}_k + \mathbf{x}_t. \tag{7}$$

Because it can be expressed as a linear operation, the viewpoint transformation can be estimated easily from a set of feature pairings. Given a model feature at $\mathbf{b}_j$ and an image feature at $\mathbf{b}_k$, the transformation aligning the two features can be obtained as the solution to the system of linear equations $\mathbf{b}_j = T(\mathbf{b}_k)$. With additional feature pairings, the problem of estimating the transformation becomes over-constrained; then the solution that is optimal in the least squares sense can be found by least squares estimation. We shall describe a solution method in section 5.3.

5 Matching and recognition methods

Recognition requires finding a consistent set of pairings between some model features and some image features, plus a viewpoint transformation that brings the paired features into close correspondence. Identifying good matches requires searching among many possible combinations of pairings and transformations. Although the positions, attributes, and relations of features provide constraints for narrowing this search, a complete search is still

impractical. Instead the goal is to order the search so that it is likely to find good matches sooner rather than later, stopping when an adequate match has been found or when many of the most likely candidates have been examined. Information about feature uncertainty can help by determining which model features to search for first, over what size image neighbourhoods to search for them, and how much to allow each to influence an estimate of the viewpoint transformation.

5.1 Match quality measure

A match is a consistent set of pairings between some model and image features, plus a transformation closely aligning paired features. We seek a match that maximizes both the number of features paired and the similarity of paired features.

Pairings are represented by $E = \langle e_1, e_2, \ldots \rangle$, where $e_j = k$ if model feature j matches image feature k, and $e_j = \perp$ if it matches nothing. H denotes the hypothesis that the modeled view of the object is present in the image. Match quality is associated with the probability of H given a set of pairings E and a viewpoint transformation T, which Bayes' theorem lets us write as

$$\mathrm{P}(H \mid E, T) = \frac{\mathrm{P}(E \mid T, H)\,\mathrm{P}(T \mid H)}{\mathrm{P}(E \wedge T)}\mathrm{P}(H) \ . \tag{8}$$

There is no practical way to represent the high-dimensional, joint probability functions $\mathrm{P}(E \mid T, H)$ and $P(E \wedge T)$ so we approximate them by adopting simplifying assumptions of feature independence. The joint probabilities are decomposed into products of low-dimensional, marginal probability functions, one per feature:

$$\mathrm{P}(H \mid E, T) \approx \prod_j \frac{\mathrm{P}(e_j \mid T, H)}{\mathrm{P}(e_j)} \frac{\mathrm{P}(T \mid H)}{\mathrm{P}(T)}\mathrm{P}(H) \ . \tag{9}$$

The measure is defined using log-probabilities to simplify calculations. Moreover, all positions of a modeled view within an image are assumed equally likely, so $\mathrm{P}(T \mid H) = \mathrm{P}(T)$. With these simplifications the measure becomes

$$g(E, T) = \log \mathrm{P}(H) + \sum_j \log \mathrm{P}(e_j \mid T, H) - \sum_j \log \mathrm{P}(e_j) \ .$$

$\mathrm{P}(H)$, the prior probability that the object as modeled is present in the image, can be estimated from the proportion of training images used to construct the model. The remaining terms are described using the following notation for random events: $\tilde{e}_j = k$, the event that model feature j matches image feature k; $\tilde{e}_j = \perp$, the event that it matches nothing; $\tilde{\mathbf{a}}_j = \mathbf{a}$, the event that it matches a feature whose attributes are $\mathbf{a}$; and $\tilde{\mathbf{b}}_j = \mathbf{b}$, the event that it matches a feature whose position, in model coordinates, is $\mathbf{b}$.

There are two cases to consider in estimating the conditional probability, $\mathrm{P}(e_j \mid T, H)$, for a model feature j.

1. When j is unmatched, this probability is estimated by considering how often j was found during training. We use a Bayesian estimator, a uniform prior, and the $\bar{m}$ and $\bar{m}_j$ statistics recorded by the model:

$$\mathrm{P}(\tilde{e}_j = \perp \mid T, H) = 1 - \mathrm{P}(\tilde{e}_j \neq \perp \mid T, H) \approx 1 - \frac{\bar{m}_j + 1}{\bar{m} + 2} \ . \tag{10}$$

2. When j is matched to image feature k, this probability is estimated by considering how often j matched an image feature during training, and how the attributes and position of k compare with those of previously matching features:

$$\begin{aligned} \mathrm{P}(\tilde{e}_j = k \mid T, H) \ \approx \ & \mathrm{P}(\tilde{e}_j \neq \perp \mid T, H)\, \mathrm{P}(\tilde{\mathbf{a}}_j = \mathbf{a}_k \mid \tilde{e}_j \neq \perp, H) \\ & \mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H) \ . \end{aligned} \tag{11}$$

$\mathrm{P}(\tilde{e}_j \neq \perp)$ is estimated as in (10). $\mathrm{P}(\tilde{\mathbf{a}}_j = \mathbf{a}_k)$ is estimated using the series of attribute vectors $\bar{A}_j$ recorded with model feature j, and a non-parametric density estimator described in (Pope, 1995). Estimation of $\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k))$, the probability that model feature j will match an image feature at position $\mathbf{b}_k$ with transformation T, is described in Sect. 5.2.

Estimates of the prior probabilities are based, in part, on measurements from a collection of images typical of those in which the object will be sought. From this collection we obtain prior probabilities of encountering various types of features with various attribute values. Prior distributions for feature positions assume a uniform distribution throughout a bounded region of model coordinate space.

5.2 Estimating feature match probability

The probability that a model and image feature match depends, in part, on their positions and on the aligning transformation. This dependency is represented by the $\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \ldots)$ term in (11). To estimate it, we transform the image feature's position into model coordinates, and then compare it with the model feature's position (Fig. 2). This comparison considers the uncertainties of the positions and transformation, which are characterized by Gaussian PDFs.

Image feature k's position is reported by its feature detector as a Gaussian PDF in $xy\theta s$ image coordinates with mean $\mathbf{b}_k^{xy\theta s}$ and covariance matrix $\mathbf{C}_k^{xy\theta s}$. To allow its transformation into model coordinates, this PDF is re-expressed in $xyuv$ image coordinates using an approximation adequate for small θ and s variances. The approximating PDF has a mean, $\mathbf{b}_k^{xyuv}$, at the same position as $\mathbf{b}_k^{xy\theta s}$, and a covariance matrix $\mathbf{C}_k^{xyuv}$ that aligns the Gaussian envelope radially, away from the $[u\ v]$ origin:

$$\begin{aligned} \mathbf{b}_k^{xyuv} &= [x_k \ \ y_k \ \ s_k \cos\theta_k \ \ s_k \sin\theta_k] \text{ and} \\ \mathbf{C}_k^{xyuv} &= \mathbf{R} \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 \\ 0 & 0 & \sigma_s^2 & 0 \\ 0 & 0 & 0 & \sigma_\theta^2 \end{bmatrix} \mathbf{R}^{\mathrm{T}} \ , \end{aligned}$$

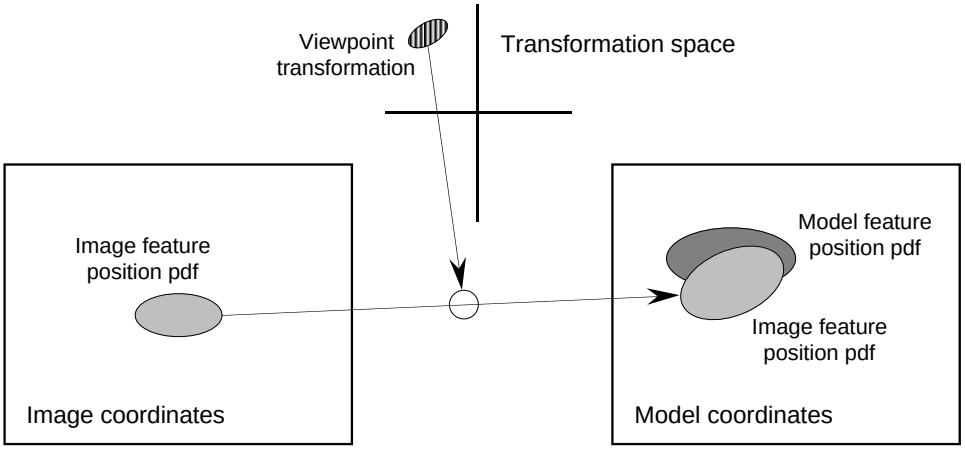


Figure 2: Comparison of image and model feature positions. An image feature's position is transformed from image coordinates (*left*) to model coordinates (*right*) according to an estimate of the viewpoint transformation. Uncertainty in the positions and the transformation are characterized by Gaussian distributions that are compared in the model coordinate space.

$$\text{where } \mathbf{R} \;=\; \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\theta_k & -\sin\theta_k \\ 0 & 0 & \sin\theta_k & \cos\theta_k \end{bmatrix}$$

and σ_l^2, σ_s^2 and σ_θ^2 are the variances in image feature position, scale and orientation estimates.

T is characterized by a Gaussian PDF over $[x_t\ y_t\ u_t\ v_t]$ vectors, with mean $\mathbf{t}$ and covariance $\mathbf{C}_t$ estimated from feature pairings as described in Sect. 5.4. Using it to transform the image feature position from $xyuv$ image to model coordinates again requires an approximation. If we would disregard the uncertainty in T, we would obtain a Gaussian PDF in model coordinates with mean $\mathbf{A}_k\mathbf{t}$ and covariance $\mathbf{A}_t\mathbf{C}_k\mathbf{A}_t^{\mathrm{T}}$. Alternatively, disregarding the uncertainty in k's position gives a Gaussian PDF in model coordinates with mean $\mathbf{A}_k\mathbf{t}$ and covariance $\mathbf{A}_k\mathbf{C}_t\mathbf{A}_k^{\mathrm{T}}$. With Gaussian PDFs for both feature position and transformation, however, the transformed position's PDF is not of Gaussian form. At best we can approximate it as such, which we do with a mean and covariance given in $xyuv$ coordinates by

$$\begin{aligned} \mathbf{b}_{kt}^{xyuv} &= \mathbf{A}_k\mathbf{t} \text{ and} \\ \mathbf{C}_{kt}^{xyuv} &= \mathbf{A}_t\mathbf{C}_k^{xyuv}\mathbf{A}_t^{\mathrm{T}} + \mathbf{A}_k\mathbf{C}_t\mathbf{A}_k^{\mathrm{T}} \ . \end{aligned}$$

Model feature j's position is also described by a Gaussian PDF in $xyuv$ model coordinates. Its mean $\mathbf{b}_j$ and covariance $\mathbf{C}_j$ are estimated from the series of position vectors $\bar{B}_j$ recorded by the model.

The desired probability (that j matches k according to their positions and the transformation) is estimated by integrating, over all $xyuv$ model coordinate positions $\mathbf{r}$, the

probability that both the transformed image feature is at $\mathbf{r}$ and the model feature matches something at $\mathbf{r}$:

$$\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \ldots) = \int_{\mathbf{r}} \mathrm{P}(\tilde{\mathbf{r}}_j = \mathbf{r})\, \mathrm{P}(\tilde{\mathbf{r}}_{kt} = \mathbf{r})\, \mathrm{d}\mathbf{r} \quad .$$

Here $\tilde{\mathbf{r}}_j$ and $\tilde{\mathbf{r}}_{kt}$ are random variables drawn from the Gaussian distributions $\mathrm{N}(\mathbf{b}_j, \mathbf{C}_j)$ and $\mathrm{N}(\mathbf{b}_{kt}, \mathbf{C}_{kt})$. It would be costly to evaluate this integral by sampling it at various $\mathbf{r}$, but fortunately the integral can be rewritten as a Gaussian since it is essentially one component in a convolution of two Gaussians:

$$\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \ldots) = G(\mathbf{b}_j - \mathbf{b}_{kt}, \mathbf{C}_j + \mathbf{C}_{kt}) \quad ,$$

where $G(\mathbf{x}, \mathbf{C})$ is a Gaussian with zero mean and covariance $\mathbf{C}$. In this form, the desired probability is easily computed.

5.3 Matching procedure

Recognition and learning require the ability to find at match between a model graph and an image graph that maximizes the match quality measure. It does not seem possible to find an optimal match through anything less than exhaustive search. Nevertheless, good matches can usually be found quickly by a procedure that combines qualities of both graph matching and iterative alignment.

5.3.1 Probabilistic alignment

To choose the initial pairings, possible pairings of high level features are rated according to the contribution each would make to the match quality measure. The pairing $\langle j, k \rangle$ receives the rating

$$g_j(k) \quad = \quad \max_T \ \log \mathrm{P}(\tilde{e}_j = k \mid T, H) - \log \mathrm{P}(\tilde{e}_j = k). \tag{12}$$

This rating favors pairings in which j has a high likelihood of matching, j and k have similar attribute values, and the transformation estimate obtained by aligning j and k has low variance. The maximum over T is easily computed because $\mathrm{P}(\tilde{e}_j = k \mid T, H)$ is a Gaussian in T.

Alignments are attempted from these initial pairings in order of decreasing rank. Each alignment begins by estimating a transformation from the initial pairing, and then proceeds by repeatedly identifying additional consistent pairings, adopting the best, and updating the transformation estimate with them until the match quality measure cannot be improved further. At this stage, pairings are selected according to how each might improve the match quality measure; thus $\langle j, k \rangle$ receives the rating

$$g_j(k; E, T) \quad = \quad \log \mathrm{P}(\tilde{e}_j = k \mid T, H) - \log \mathrm{P}(\tilde{e}_j = k). \tag{13}$$

This favors the same qualities as equation 12 while also favoring pairings that are aligned closely by the estimated transformation. In order for $\langle j, k \rangle$ to be adopted, it must rate at least as well as the alternative of leaving j unmatched, which receives the rating

$$g_j(\perp; E, T) \;\; = \;\; \log \mathrm{P}(\tilde{e}_j = \perp | \; T, H) - \log \mathrm{P}(\tilde{e}_j = \perp). \tag{14}$$

Significant computation is involved in rating and ranking the pairings needed to extend an alignment. Consequently, pairings are adopted in batches so that this computation need only be done infrequently. Moreover, in the course of an alignment, batch size is increased as the transformation estimate is further refined so that each batch can be made as large as possible. A schedule that seems to work well is to start an alignment with a small batch of pairings (we use five), and to double the batch size with each batch adopted.

5.4 Estimating the aligning transformation

From a series of feature pairings, an aligning transformation is estimated by finding the least-squares solution to a system of linear equations. Each pairing $\langle j, k \rangle$ contributes to the system the equations

$$\mathbf{U}_j^{-1} \, \mathbf{A}_k \, \mathbf{t} = \mathbf{U}_j^{-1} \, \mathbf{b}_j + \tilde{\mathbf{e}} \;\; .$$

$\mathbf{A}_k$ is the matrix representation of image feature k's mean position, $\mathbf{t} = [x_t \; y_t \; u_t \; v_t]$ is the transformation estimate, and $\mathbf{b}_j$ is model feature j's mean position. $\mathbf{U}_j$ is the upper triangular square root of j's position covariance (i.e., $\mathbf{C}_j = \mathbf{U}_j \mathbf{U}_j^{\mathrm{T}}$); it weights both sides of the equation so that the residual error $\tilde{\mathbf{e}}$ has unit variance.

A recursive estimator solves the system, efficiently updating the transformation estimate as pairings are adopted. We use the square root information filter (SRIF) (Bierman, 1977) form of the Kalman filter for its numerical stability, and its efficiency with batched measurements. The SRIF works by updating the square root of the information matrix, which is the inverse of the estimate's covariance matrix. The initial square root, $\mathbf{R}_1$, and state vector, $\mathbf{z}_1$, are obtained from the first pairing $\langle j, k \rangle$ by

$$\mathbf{R}_1 = \mathbf{U}_j^{-1} \, \mathbf{A}_k \;\; \text{and} \;\; \mathbf{z}_1 = \mathbf{U}_j^{-1} \, \mathbf{b}_j \;\; .$$

With each subsequent pairing $\langle j, k \rangle$, the estimate is updated by triangularizing a matrix composed of the previous estimate and data from the new pairing:

$$\begin{bmatrix} \mathbf{R}_{i-1} & \mathbf{z}_{i-1} \\ \mathbf{U}_j^{-1}\mathbf{A}_k & \mathbf{U}_j^{-1}\mathbf{b}_j \end{bmatrix} \overset{\triangle}{\rightarrow} \begin{bmatrix} \mathbf{R}_i & \mathbf{z}_i \\ 0 & \mathbf{e}_i \end{bmatrix} \;\; .$$

When needed, the transformation and its covariance are obtained from the triangular $\mathbf{R}_i$ by back substitution:

$$\mathbf{t}_i = \mathbf{R}_i^{-1} \mathbf{z}_i \;\; \text{and} \;\; \mathbf{C}_{t_i} = \mathbf{R}_i^{-1} \mathbf{R}_i^{-\mathrm{T}} \;\; .$$

5.5 Verification

Once a match has been found between a model graph and an image graph, it must be decided whether the match represents an actual instance of the modeled object in the image. A general approach to this problem would use decision theory to weigh prior expectations, evidence derived from the match, and the consequences of an incorrect decision. However, we will use a simpler approach that only considers the number and type of matching features and the accuracy with which they match.

The match quality measure used to guide matching provides one indication of a match's significance. A simple way to accept or reject matches, then, might be to require that this measure exceeds some threshold. However, the measure is unsuitable for this use because its range differs widely among objects according to what high-level features they have. High-level features that represent groupings of low-level ones violate the feature independence assumption; consequently, the match quality measure is biased by an amount that depends on what high-level features are present in the model. Whereas this bias seems to have no adverse effect on the outcome of matching any one model graph, it makes it difficult to establish a single threshold for testing the match quality measure of any model graph. Thus the verification method we present here considers only the lowest-level features of the model graph—those that do not group any other model graph features.

When counting paired model features, we weight each one according to its likelihood of being paired, thereby assigning greatest importance to the features that contribute most to the likelihood that the object is present. For model feature j, the likelihood of being paired,

$$\mathrm{L}(\tilde{e}_j \neq \perp |\, T, H) \;=\; \frac{\mathrm{P}(\tilde{e}_j \neq \perp |\, T, H)}{\mathrm{P}(\tilde{e}_j \neq \perp)},$$

is estimated using statistics recorded for feature j.

The count of each model feature is also weighted according to how well it is fit by image features. When j is a curve segment, this weighting component is based on the fraction of j matched by nearby image curve segments. The fraction is estimated using a simple approximation: The lengths of image curve segments matching j are totaled, the total length is transformed into model coordinates, and the transformed value is divided by the length of j. With s_t denoting the scaling component of the viewpoint transformation T, and s_j and s_k denoting the lengths of j and k in model and image coordinates, respectively, the fraction of j covered by image curve segments is defined as

$$\mathsf{Support}(j; E, T) \;=\; \min\left(1,\, \frac{s_t}{s_j} \sum_{\langle j,k\rangle \in E} s_k\right), \quad j \text{ a curve segment.}$$

If we were to accept matches that paired a fixed number of model features regardless of model complexity, then with greater model complexity we would have an increased likelihood of accepting incorrect matches. For example, requiring that ten model features be paired may make sense for a model of twenty features, but for a model of a thousand features, any incorrect match is likely to contain at least that many "accidental" pairings.

Thus we have chosen instead to require that some minimum fraction of the elements of C be paired. We define this fraction as

$$\mathsf{Support}(E,T) \;=\; \frac{\sum_{j\in C} \mathrm{L}(\tilde{e}_j \neq \perp |\, T,H)\, \mathsf{Support}(j,E,T)}{\sum_{j\in C} \mathrm{L}(\tilde{e}_j \neq \perp |\, T,H)}. \tag{15}$$

A match $\langle E,T\rangle$ is accepted if $\mathsf{Support}(E,T)$ achieves a certain threshold τ. To validate this verification method and to determine a suitable value for τ, we have measured the distribution of $\mathsf{Support}(E,T)$ for correct and incorrect matches between various model graphs and their respective training image graphs. The distributions are well separated, with most correct matches achieving $\mathsf{Support}(E,T) > 0.6$ and most incorrect matches achieving $\mathsf{Support}(E,T) < 0.3$.

6 Model learning procedure

The learning procedure assembles one or more model graphs from a series of training images showing various views of an object. To do this, it clusters the training images into groups and constructs model graphs generalizing the contents of each group. We shall describe first the clustering procedure, and then the generalization procedure, which the clustering procedure invokes repeatedly.

We use $\mathcal{X}$ to denote the series of training images for one object. During learning, the object's model $\mathcal{M}$ consists of a series of clusters $\mathcal{X}_i \subseteq \mathcal{X}$, each with an associated model graph $\bar{G}_i$. Once learning is complete, only the model graphs must be retained to support recognition.

6.1 Clustering training images

An incremental conceptual clustering algorithm is used to create clusters among the training images. Clustering is incremental in that, as each training image is acquired, it is assigned to an existing cluster or used to form a new one. Like other conceptual clustering algorithms, such as COBWEB (Fisher, 1987), the algorithm uses a global measure of overall clustering quality to guide clustering decisions. This measure is chosen to promote and balance two somewhat-conflicting qualities. On one hand, it favors clusterings that result in simple, concise, and efficient models, while on the other hand, it favors clusterings whose resulting model graphs accurately characterize the training images.

The minimum description length principle (Rissanen, 1983) is used to quantify and balance these two qualities. The principle suggests that the learning procedure choose a model that minimizes the number of symbols needed to encode first the model and then the training images. It favors simple models as they can be encoded concisely, and it favors accurate models as they allow the training images to be encoded concisely once the model has been provided. The clustering quality measure to be minimized is defined as $L(\mathcal{M}) + L(\mathcal{X} \mid \mathcal{M})$, where $L(\mathcal{M})$ is the number of bits needed to encode the model $\mathcal{M}$,

and $L(\mathcal{X} \mid \mathcal{M})$ is the number of bits needed to encode the training images $\mathcal{X}$ when $\mathcal{M}$ is known.

To define $L(\mathcal{M})$ we specify a coding scheme for models that concisely enumerates each of a model's graphs along with its nodes, arcs, attribute vectors and position vectors (see (Pope, 1995) for full details of the coding scheme). Then $L(\mathcal{M})$ is simply the number of bits needed to encode $\mathcal{M}$ according to this scheme.

To define $L(\mathcal{X} \mid \mathcal{M})$ we draw on the fact that given any probability distribution $\mathrm{P}(x)$, there exists a coding scheme, the most efficient possible, that achieves essentially $L(x) = -\log_2 \mathrm{P}(x)$. Recall that the match quality measure is based on an estimate of the probability that a match represents a true occurrence of the modeled object in the image. We use this probability to estimate $\mathrm{P}(X \mid \bar{G}_i)$, the probability that the appearance represented by image X may occur according to the appearance distribution represented by model graph $\bar{G}_i$:

$$\mathrm{P}(X \mid \bar{G}_i) \;=\; \max_{\langle E,T \rangle} \mathrm{P}(H \mid E, T) \;\; .$$

This probability can be computed for any given image graph X and model graph G_i, using the matching procedure (Sect. 5.3) to maximize $\mathrm{P}(H \mid E, T)$ over matches $\langle E, T \rangle$. $\mathrm{P}(X \mid \bar{G}_i)$ is then used to estimate the length of an encoding of X given $\bar{G}_i$:

$$L(X \mid \bar{G}_i) \;=\; \min_{\langle E,T \rangle} \Big(-\log_2 \mathrm{P}(X \mid \bar{G}_i) + L_u(X, E) \Big) \;\; .$$

The $L_u(X, E)$ term is the length of an encoding of unmatched features of X, which we define using a simple coding scheme comparable to that used for model graphs. Finally, we define $L(\mathcal{X} \mid \mathcal{M})$ by assuming that for any $X \in \mathcal{X}_i \subseteq \mathcal{X}$, the best match between X and any $\bar{G}_j \in \mathcal{M}$ will be that between X and $\bar{G}_i$ (the model graph obtained by generalizing the group containing X). Then the length of the encoding of each $X \in X$ in terms of the set of model graphs $\mathcal{M}$ is the sum of the lengths of the encodings of each in terms of its respective model graph:

$$L(\mathcal{X} \mid \mathcal{M}) \;=\; \sum_i \sum_{X \in \mathcal{X}_i} L(X \mid \bar{G}_i) \;\; .$$

As each training image is acquired it is assigned to an existing cluster or used to form a new one. Choices among clustering alternatives are made to minimize the resulting $L(\mathcal{M}) + L(\mathcal{X} \mid \mathcal{M})$. When evaluating an alternative, each cluster's subset of training images $\mathcal{X}_i$ is first generalized to form a model graph $\bar{G}_i$ as described below.

6.2 Generalizing training images

Within each cluster, training images are merged to form a single model graph that represents a generalization of those images. An initial model graph is formed from the first training image's graph. That model graph is then matched with each subsequent training image's graph and revised after each match according to the match result. A model feature j that matches an image feature k receives an additional attribute vector $\mathbf{a}_k$ and position $\mathbf{b}_k$

for its series $\bar{A}_j$ and $\bar{B}_j$. Unmatched image features are used to extend the model graph, while model features that remain largely unmatched are eventually pruned. After several training images have been processed in this way the model graph nears an equilibrium, containing the most consistent features with representative populations of sample attribute vectors and positions for each.

7 Experimental results

In this section we describe several experiments involving a system implemented to test our recognition learning method. This system, called OLIVER, learns to recognize 3–D objects in 2–D intensity images.

OLIVER has been implemented within the framework of Vista, a versatile and extensible software environment designed to support computer vision research (Pope and Lowe, 1994). Both OLIVER and Vista are written in C to run on UNIX workstations. The execution times reported here were measured on a Sun SPARCstation 10/51 processor.

The focus of this research has been on the model learning and representation, which is independent of the particular features used for matching. To test the approach, we have chosen to use a basic repertoire of edge-based features. While some recent approaches to recognition have been based on image pixel intensities or image derivative magnitudes, the locations of intensity discontinuities may be more robust to illumination and imaging variations. For example, the silhouette boundaries of an object on a cluttered background will have image derivatives of unknown sign and magnitude. The same is true for edges separating surfaces of different orientation under differing directions of illumination.

In the following experiments, the lowest-level features are straight, circular and elliptical edge segments. An edge curve of any shape can be represented by approximating it with a series of primitive segments. Additional higher-level features represent groupings of these, such as junctions, groups of adjacent junctions, pairs of parallel segments, and convex regions. For full details on the derivation of these features, see (Pope, 1995). In the future, this set could be augmented with other features, such as those based on image derivatives, color, or texture, but the current features are suited for a wide range of objects.

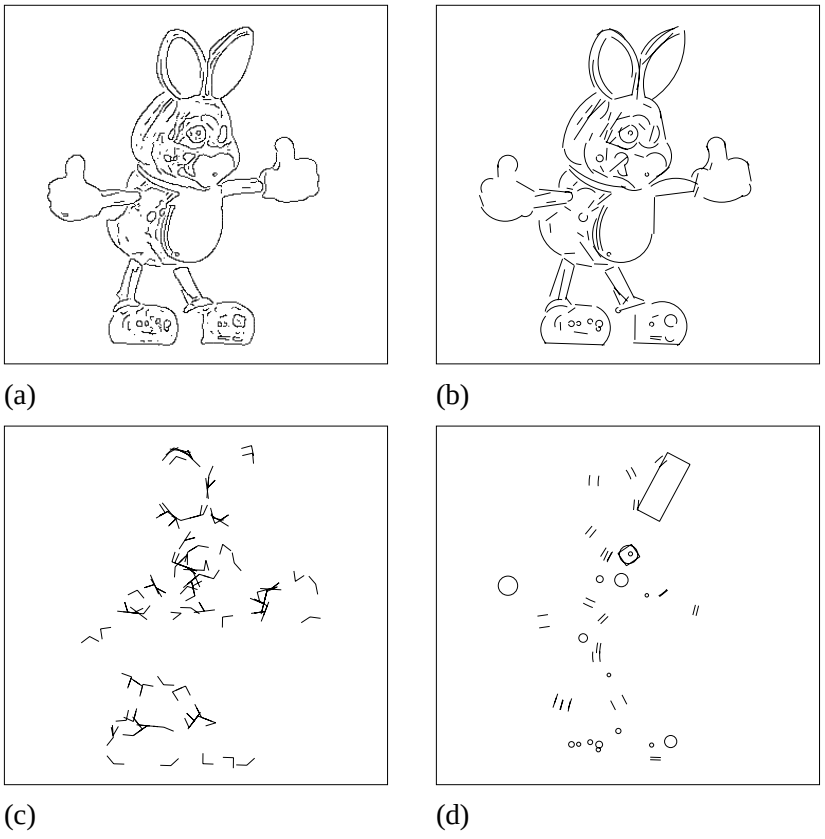
7.1 Illustrative experiment

The experiment described in this section demonstrates typical performance of the system in learning to recognize a complex object. The test object is a toy bunny shown in figure 3. Training images of the bunny were acquired at 5° increments of camera elevation and azimuth over 25° of elevation and 90° of azimuth.

Feature detection, including edge detection, curve segmentation, and grouping, required about 30 seconds of CPU time per training image (we believe that much faster grouping processes are possible, but this was not the focus of the research). Figure 4 depicts some of the features found in one image, which include 4475 edgels, 81 straight lines and 67 circular arcs.

25° elevation, 90° azimuth 0° elevation, 90° azimuth 0° elevation, 10° azimuth

Figure 3: Bunny training images. Images were acquired at 5° intervals over camera elevations of 0° to 25° and azimuths of 0° to 90°. Shown here are three of the 112 images.



(a) (b)

(c) (d)

Figure 4: Features of a bunny training image. Shown here are selected features found in the 0° elevation, 10° azimuth training image (right image in figure 3). (a) Edgels. (b) Curve features. (c) L-junction features. (d) Parallel-curve features (depicted by parallel lines), Region features (depicted by rectangles), and Ellipse features (depicted by circles).

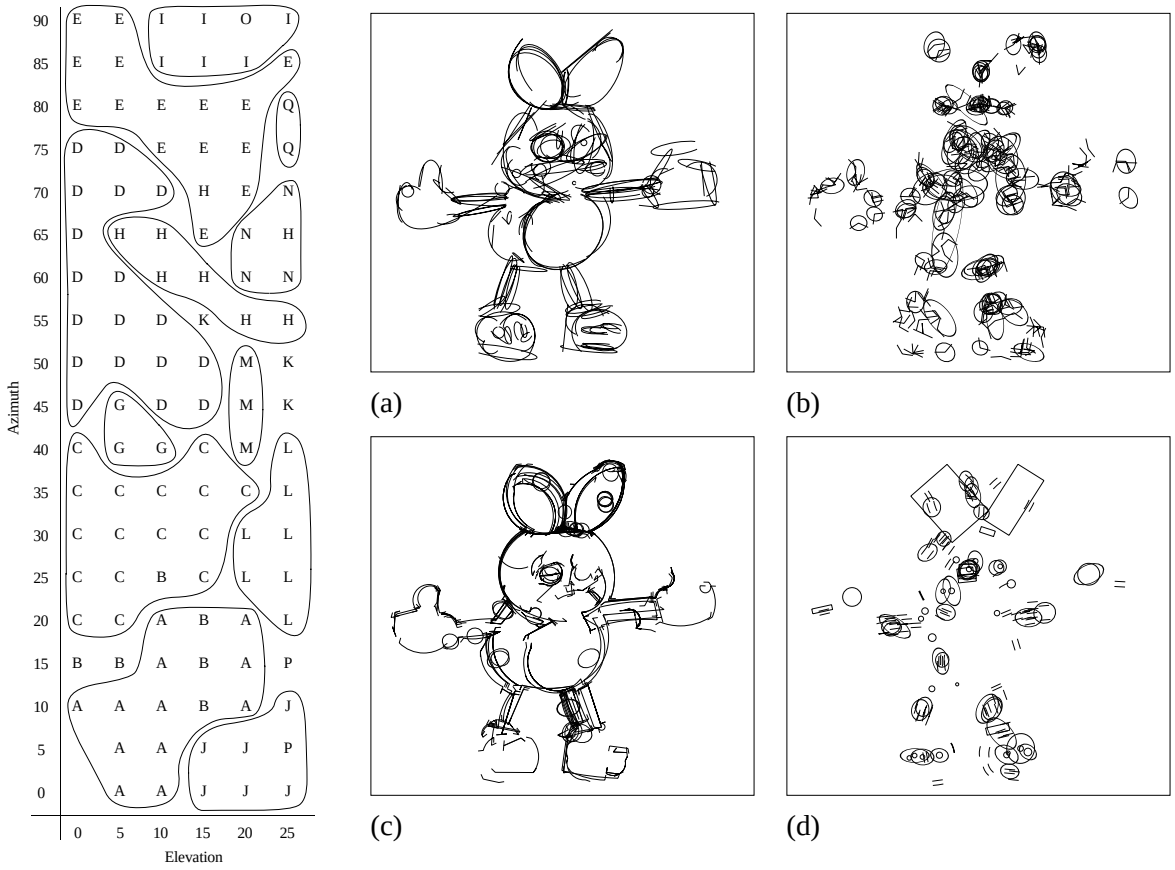


Figure 5: On the left are the training image clusters. Seventeen clusters, designated A through Q, were formed from the 112 training images. Contours delineate the approximate scope of the view classes associated with some of the clusters. On the right are selected features of the model graph obtained by generalizing the training images assigned to cluster C. Each feature is drawn at its mean location. (a) Curve features. (b) L-junction features. (c) Connected edge features. (d) Parallel curve, Region and Ellipse features.

During the first phase of clustering, the system divided the training images among 19 clusters. In the second phase, it reassigned two training images that remained the sole members of their clusters, leaving the 17 clusters shown in figure 5. Because this object's appearance varies smoothly with changes in viewpoint across much of the viewing range, it is not surprising that the clusters generally occupy contiguous regions of the viewsphere.

When training images were presented to the system in other sequences, the system produced different clusterings than that shown in figure 5. However, although cluster boundaries varied, qualities such as the number, extent, and cohesiveness of the clusters remained largely unaffected.

As this is a one-time batch operation, little effort was devoted to optimizing efficiency. Altogether, 19.4 hours of CPU time were required to cluster the training images and to induce a model graph generalization of each cluster.

Figure 5 shows features of the model graph representing cluster C. Ellipses are drawn

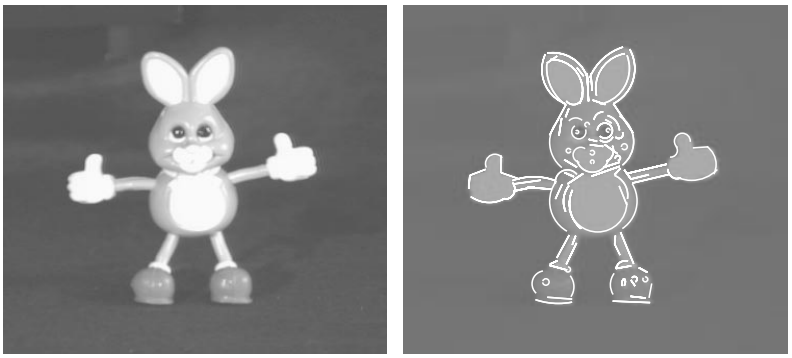


Figure 6: Bunny test image 1. *Left:* Image. *Right:* Match of bunny model graph D with test image.

for certain features to show two standard deviations of location uncertainty. To reduce clutter and to give some indication of feature significance, they are drawn only for those features that were found in a majority of training images. Considerable variation in location uncertainty is evident. Some L-junction features have particularly large uncertainty and, consequently, they will be afforded little importance during matching.

Figure 7 reports the results of matching the image graph for test image 1 with each of the bunny model graphs. For this test, match searches were allowed to examine all alignment hypotheses. Typically, there were 10–20 hypotheses examined for each pair of model and image graphs, and about five seconds of CPU time were needed to extend and evaluate each one. The matches reported here are those achieving the highest match quality measure.

The model graph generalizing cluster D provides the best match with the image graph (as judged by each match's support measure). This is to be expected as the test image was acquired from a viewpoint surrounded by that cluster's training images. Moreover, other model graphs that match the image graph (although not as well) are all from neighbouring regions of the viewsphere. Image features included in the best match, that with model graph D, are shown in figure 6.

For test image 2, shown in figure 8, additional clutter was present. Figure 7 reports the result of matching the image graph of test image 2 with each of the bunny model graphs. This time each match search was limited to 20 alignment hypotheses and about six seconds of CPU time were needed to extend and evaluate each hypothesis. Due to the additional clutter in the image, only model graph D correctly matched the image.

This section has demonstrated the system's typical performance in learning models of complex objects, and in using those models to accomplish recognition. Figure 9 shows recognition of an even more complex object with significant occlusion.

7.2 Additional Experiments

In this section, some additional experiments are briefly described in order to illustrate certain noteworthy aspects of the system's behaviour.

Model Graph	Match with Test Image 1				Match with Test Image 2			
	Correct	Quality	Pairings	Support	Correct	Quality	Pairings	Support
A		998	164	0.35		862	191	0.34
B		596	141	0.22		726	166	0.24
C	Yes	1952	252	0.56		1634	278	0.45
D	Yes	2347	253	0.62	Yes	1502	207	0.57
E		841	160	0.33		1043	205	0.39
F		298	100	0.15		275	115	0.20
G		147	88	0.18		278	122	0.27
H	Yes	1175	201	0.41		438	139	0.29
I		356	91	0.19		879	186	0.35
J		357	87	0.17		768	197	0.29
K	Yes	912	195	0.46		422	158	0.33
L		958	175	0.38		521	165	0.25
M	Yes	1024	200	0.46		649	177	0.33
N	Yes	613	177	0.39		593	162	0.29
O		151	72	0.14		187	90	0.23
P		206	102	0.26		579	161	0.30
Q		204	86	0.19		344	116	0.31

Figure 7: Each row documents the results of matching a model graph with the two image graphs, those describing bunny test images 1 and 2. Reported for each match are the following. Correct: whether the match correctly identified most features of the object visible in the image, as judged by the experimenter. Quality: the match's match quality measure, $g(E, T)$. Pairings: the number of image features paired. Support: the match's support measure, $\mathsf{Support}(E, T)$.

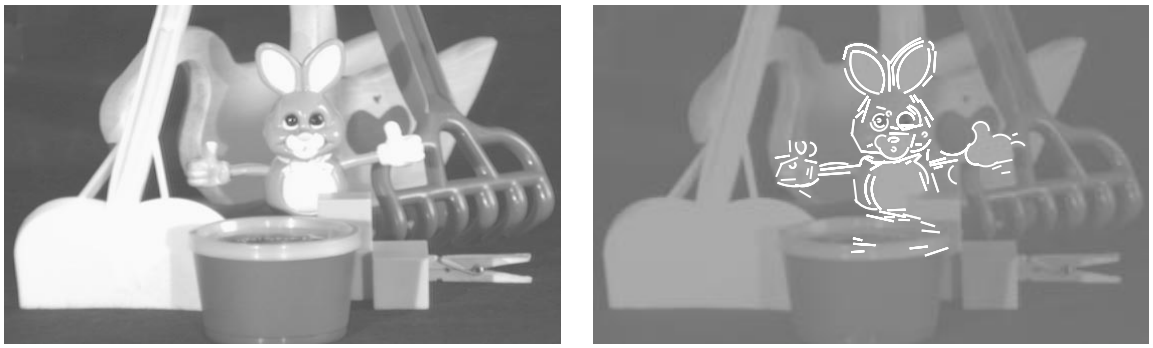


Figure 8: *Left:* Bunny test image 2 with clutter and occlusion. *Right:* Match of bunny model graph D with test image 2.

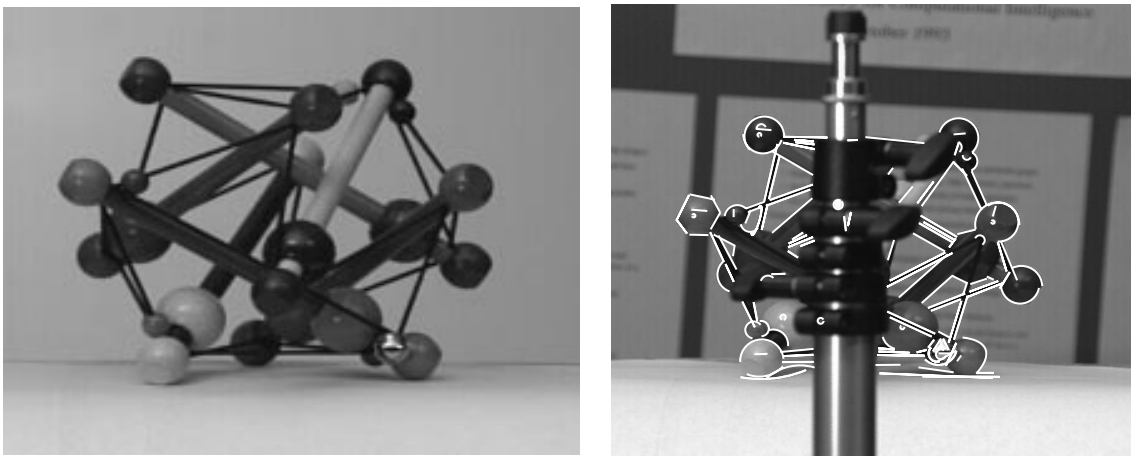


Figure 9: Example showing recognition of a complex object with substantial occlusion. *Left:* One of several training images. *Right:* Image curve features included in the match.

12° elevation, 60° azimuth 12° elevation, 0° azimuth 0° elevation, 0° azimuth

Figure 10: Shoe training images. Images were acquired at 6° intervals over camera elevations of 0° to 12° and azimuths of 0° to 60°. Shown here are three of the 33 images.

7.2.1 Effects of feature distribution

When some regions of an object are much richer in stable features than others, those regions can dominate the matching processes that underlie learning and recognition. For example, most features of the shoe shown in figure 10 are concentrated near its centre. Moreover, as the shoe rotates about its vertical axis, features near the shoe's centre shift by small amounts while those near its heel and toe undergo much larger changes. Thus, when training images of the shoe are clustered during model learning, the many stable features near the shoe's centre are used to match training images over a large range of rotation, while the few variable features defining the heel and toe are dropped as being unreliable. The result is a model graph, like that shown in figure 11 (left), with relatively few features defining the shoe's extremities.

If the dropped features are deemed important, we can encourage the system to retain them in the models it produces by setting a higher standard for acceptable matches. For example, requiring a higher Support measure ensures that matches will include more of an object's features. Thus, fewer of those features will be judged unreliable and more will be retained by the model. Figure 11 (right) shows a model graph that was produced with a Support threshold of 0.6 rather than the usual value of 0.5; it provides somewhat more

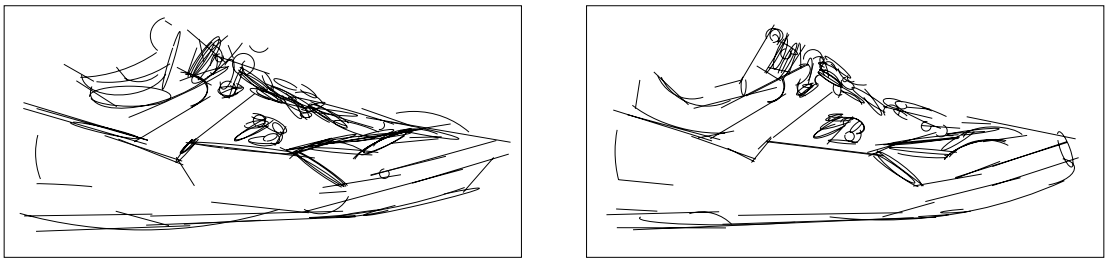


Figure 11: *Left:* Curve features for a model that generalizes 14 training images (support threshold of 0.5). *Right:* Model that generalizes 7 training images (support threshold of 0.6).

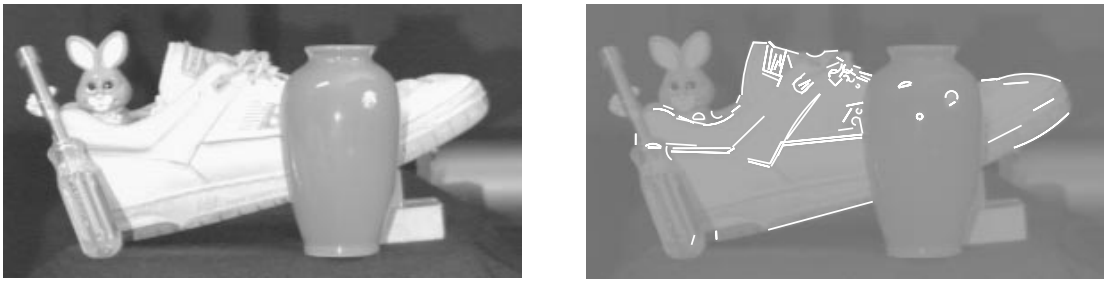


Figure 12: Shoe recognition example. *Left:* Test image. *Right:* Image curve features included in a match to shoe model.

accurate representation of the shoe's heel and toe. Figure 12 shows this model graph being used for recognition.

7.2.2 Articulate objects

Just as the system will use multiple views to model an object's appearance over a range of viewpoints, it will use additional views to model a flexible or articulate object's appearance over a range of configurations. In general, the number of views needed increases exponentially with the number of dimensions along which the object's appearance may vary. This could presumably be addressed by a part-based modeling and clustering approach that separated the independent model parts.

The toy boat shown in figure 13 has a sail that rotates about the boat's vertical axis. Training images were acquired at camera elevations of $0°$, $5°$, and $10°$; camera azimuths of $0°$, $5°$, ..., $35°$; and sail angles of $0°$, $10°$, ..., $40°$. The system's learning procedure clustered these 120 images to produce 64 model views. Features of two of the model graphs are shown in figure 14. In comparison, only 13 views were needed to cover the same range of viewpoints when the sail angle was kept fixed at $0°$.

0° elevation, 0° azimuth, 0° sail angle

0° elevation, 0° azimuth, 40° sail angle

10° elevation, 0° azimuth, 0° sail angle

Figure 13: Boat training images. Images were acquired at elevations of 0°, 5°, and 10°; azimuths of 0°, 5°, ..., 35°; and sail angles of 0°, 10°, and 20°. Shown here are three of the 120 images.

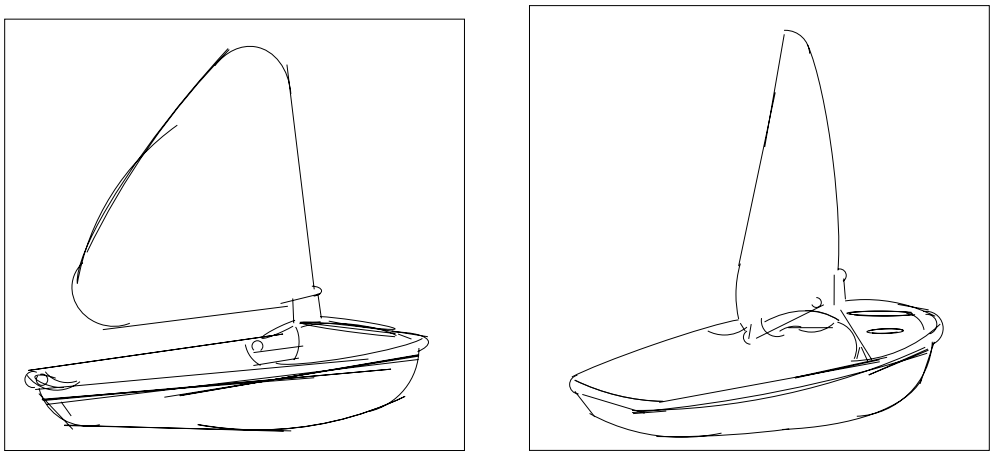


Figure 14: Boat model graphs. Shown here are curve features of model graphs that have been generalized from two clusters of boat training images.

8 Conclusion

We have presented a method of modeling the appearance of objects, of automatically acquiring such models from training images, and of using the models to accomplish recognition. This method can handle complex, real-world objects. In principle, it can be used to recognize any object by its appearance, provided it is given a sufficient range of training images, sufficient storage for model views, and an appropriate repertoire of feature types.

The main features of the method are as follows:

(a) Objects are modeled in terms of their appearance, rather than shape, to avoid any need to model the image formation process. This allows unusually complex objects to be modeled and recognized efficiently and reliably.

(b) Appearance is described using discrete features of various types, ranging widely in scale, complexity, and specificity. This repertoire can be extended considerably, still within the framework of the approach, to accommodate a large variety of objects.

(c) An object model represents a probability distribution over possible appearances of the object, assigning high probability to the object's most likely manifestations. Thus, learning an object model from training images amounts to estimating a distribution from a representative sampling of that distribution.

(d) A match quality measure provides a principled means of evaluating a match between a model and an image. It combines probabilities that are estimated using distributions recorded by the model. The measure leads naturally to an efficient matching procedure, probabilistic alignment, used to accomplish both learning and recognition.

(e) The model learning procedure has two components. One component identifies clusters of training images that ought to correspond to distinct model views. It does so by maximizing a measure that, by application of the minimum description length principle, combines the qualities of model simplicity and accuracy. The second component induces probabilistic generalizations of the images within each cluster. Working together, the two components construct a model by clustering training images, and, within each cluster, generalizing the images to form a model view.

8.1 Topics for further research

Modeling a multifarious or highly flexible object with this approach may require an impractically large number of model views. For these objects, a more effective strategy may be first to recognize parts, and then to recognize the whole object as a configuration of those parts. The present method could perhaps be extended to employ this strategy by assigning parts the role of high level features.

Speed in both learning and recognition tasks could be greatly improved by the addition of an indexing component, which would examine image features and suggest likely model views for the matching procedure to consider. Existing indexing methods (Beis and Lowe, 1999) could be used, with the attribute vectors of high-level features serving as index keys. Of course, more efficient methods for feature detection would also be important.

Extending the feature repertoire would allow the method to work more effectively with a broader class of objects. It would be useful to have features representing additional groupings of intensity edges, such as symmetric arrangements and repeated patterns, and features representing local image regions with color or texture properties.

Some challenging issues remain regarding how to organize a large collection of acquired models for greater efficiency. Savings in both storage and recognition time could be achieved by identifying parts or patterns common to several objects, factoring those parts out of their respective models, and recognizing the parts individually prior to recognizing their aggregates. Associating new feature types with some of the common parts and patterns would provide a means of automatically extending the feature repertoire and adapting it to the objects encountered during training. Furthermore, the same techniques of

identifying and abstracting parts could be used to decompose flexible objects into simpler components, allowing those objects to be modeled with fewer views.

Acknowledgments

The authors would like to thank Jim Little, Bob Woodham, and Alan Mackworth for their ongoing comments on this research. This research was sponsored by the Natural Sciences and Engineering Research Council of Canada (NSERC) and through the Institute for Robotics and Intelligent Systems (IRIS) Network of Centres of Excellence.

References

Ayache, N. and O. D. Faugeras (1986). HYPER: A new approach for the recognition and positioning of two-dimensional objects. *IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-8*(1), 44–54.

Beis, J. S. and D. G. Lowe (1999). Indexing without invariants in 3D object recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence 21*(10), 1000–1015.

Bierman, G. J. (1977). *Factorization Methods for Discrete Sequential Estimation*. Academic Press.

Bolles, R. C. and R. A. Cain (1982). Recognizing and locating partially visible objects: The local-feature-focus method. *Int. J. of Robotics Research 1*(3), 57–82.

Breuel, T. M. (1992). *Geometric Aspects of Visual Object Recognition*. Ph. D. thesis, Mass. Inst. Technol.

Brooks, R. A. (1981). Symbolic reasoning among 3-D models and 2-D images. *Artificial Intelligence 17*, 285–348.

Burl, M. C., M. Weber, and P. Perona (1998). A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. European Conf. on Computer Vision*, pp. 628–641.

Burns, J. B. and E. M. Riseman (1992). Matching complex images to multiple 3D objects using view description networks. In *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 328–334.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning 2*, 139–172.

Grimson, W. E. L. and T. Lozano-Pérez (1987). Localizing overlapping parts by searching the interpretation tree. *IEEE Trans. Pattern Analysis and Machine Intelligence 9*(4), 469–482.

Hel-Or, Y. and M. Werman (1995). Pose estimation by fusing noisy data of different dimensions. *IEEE Trans. Pattern Analysis and Machine Intelligence 17*(2), 195–201.

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

Lowe, D. G. (1985). *Perceptual Organization and Visual Recognition*. Kluwer.

Lowe, D. G. (1987). Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence 31*, 355–395.

Murase, H. and S. K. Nayar (1995). Visual learning and recognition of 3D objects from appearance. *International Journal of Computer Vision 14*(1), 5–24.

Nelson, R. C. and A. Selinger (1998). A cubist approach to object recognition. In *Proc. Int. Conf. Computer Vision*, pp. 614–621.

Petitjean, S., J. Ponce, and D. J. Kriegman (1992). Computing exact aspect graphs of curved objects: Algebraic surfaces. *International Journal of Computer Vision 9*(3), 231–255.

Pope, A. R. (1995). *Learning to Recognize Objects in Images: Acquiring and Using Probabilistic Models of Appearance*. Ph. D. thesis, Computer Science Dept., Univ. of British Columbia.

Pope, A. R. and D. G. Lowe (1994). Vista: A software environment for computer vision research. In *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 768–772.

Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *Annals of Statistics 11*(2), 416–431.

Roberts, L. G. (1965). Machine perception of three-dimensional solids. In J. Tippett (ed.), *Optical and Electro-Optical Information Processing*, pp. 159–197. MIT Press.

Schiele, B. and J. Crowley (1996). Object recognition using multidimensional receptive field histograms. In *Fourth European Conference on Computer Vision*, Cambridge, UK, pp. 610–619.

Schmid, C. and R. Mohr (1997). Local grayvalue invariants for image retrieval. *IEEE Trans. Pattern Analysis and Machine Intelligence 19*(5), 530–534.

Swain, M. and D. Ballard (1991). Color indexing. *International Journal of Computer Vision 7*(1), 11–32.

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

Van Huffel, S. and J. Vandewalle (1991). *The Total Least Squares Problem: Computational Aspects and Analysis*, Volume 9 of *Frontiers in Applied Mathematics*. Philadelphia: Soc. for Industrial and Applied Mathematics.

Wells III, W. M. (1997). Statistical approaches to feature-based object recognition. *International Journal of Computer Vision 21*(1/2), 63–98.