# Modeling Positional Uncertainty in Object Recognition

Arthur R. Pope and David G. Lowe

Technical Report 94-32

November 1994

## Abstract

Iterative alignment is one method for feature-based matching of an image and a model for the purpose of object recognition. The method alternately hypothesizes feature pairings and estimates a viewpoint transformation from those pairings; at each stage a refined transformation estimate is used to suggest additional pairings.

This paper extends iterative alignment in the domain of 2D similarity transformations so that it represents the uncertainty in the position of each model and image feature, and that of the transformation estimate. A model describes probabilistically the significance, position, and intrinsic attributes of each feature, plus topological relations among features. A measure of the match between a model and an image integrates all four of these, and leads to an efficient matching procedure called *probabilistic alignment*. That procedure supports both recognition and a learning procedure for acquiring models from training images.

By explicitly representing uncertainty, one model can satisfactorily describe appearance over a wider range of viewing conditions. Thus, when models represent 2D characteristic views of a 3D object, fewer models are needed. Experiments demonstrating the effectiveness of this approach are reported.

## Contents

## 1 Introduction

Object recognition is made difficult by the way an object's appearance varies under different viewing conditions. Changes in viewpoint, changes in lighting, and flexing of the object can all cause changes in the detectability and relative positions of the object's distinguishing features. To successfully recognize the object on the basis of a stored model, that model and the process used to compare it with an image must together account for the allowable range of variation in the object's appearance.

A common approach is to model a 2D or 3D object by a series of characteristic views, each representative of the object's appearance under some small range of viewing conditions. Collectively, these characteristic views cover the full range of expected appearances of the object. Systems that take this approach generally assume that all features visible in a characteristic view have the same likelihood of being detected and the same range of variation in position.

Clearly, though, some features are more likely to be detected, can be more accurately localized, or move less with changes in viewpoint than other features. Perhaps it is because these differences are difficult to quantify that researchers have often disregarded them, assuming the same uncertainty statistics for all features. A system that can learn its models directly from training images, however, may have a simple and direct way of quantifying the uncertainty of each feature: the system can measure it during training. By observing how features differ among training images the system can estimate the detectability and positional uncertainty of each feature included in the models it acquires.

There are several ways that information about feature uncertainty, if known, ought to guide the matching process that underlies recognition. Features of the object that are most likely to be detected when the object is present, and least likely to be detected when it is absent, should be given priority during matching. Features that can be localized well should contribute most to an estimate of the object's position in an image. And features whose positions vary most should be sought over the largest neighborhoods of the image.

Our hypothesis is that feature uncertainty information can be obtained reliably from training images, and that the information can be used effectively in the manner just outlined to improve recognition performance. In this paper we describe an approach for representing, learning, and using feature uncertainty information, and experiments involving a system constructed to test that approach. The approach models 3D objects using characteristic views, each describing numerous features of various types. These characteristic view models include information about feature uncertainty learned entirely from training images.

Our method of matching a model and an image is similar to the iterative alignment method of object recognition [1, 9, 10]: we hypothesize some initial pairings between model features and image features, use those pairings to estimate viewpoint, use the viewpoint estimate to evaluate and choose additional pairings, and so on until as many features as possible have been matched. But along with the viewpoint estimate we also maintain an estimate of viewpoint uncertainty that is derived from the uncertainties of the paired model and image features. Both the viewpoint estimate and its uncertainty are used to evaluate potential feature pairings so that model features with more certain positions are paired sooner. This use of uncertainty information produces a better ordering of feature pairings, resulting in a faster search with less backtracking. The method is called *probabalistic alignment* to emphasize its use of probability theory.

## 2 Related Research

Among methods for recognizing 3D objects by matching discrete features with models of 2D characteristic views, there are generally three classes. Alignment methods, as described in the introduction, use feature pairings to estimate a transformation, and then use that transformation estimate to suggest further pairings. Ayache and Faugeras [1] showed that, when the transformation is a 2D similarity transformation represented in a certain way, it can be estimated by a recursive, linear least-squares estimator like the Kalman filter. This is particularly efficient because a transformation can be estimated directly, without search, and because the estimate can be updated with each new pairing at little cost. For these efficiency reasons we have adopted a similar formulation.

Alignment methods may consider feature position uncertainty in estimating the transformation so that the estimate is influenced most by the more precisely localized features. Moreover, uncertainty in feature positions and misalignment of matched features yields uncertainty in the transformation estimate; that uncertainty, in turn, can influence the selection of features for pairing so that more certain pairings are favored. Ayache and Faugeras used simple heuristics to estimate model feature uncertainty and to choose model features for pairing. In contrast, because our system acquires models from training images it is able to directly measure the positional variance of each model feature, maintain a meaningful estimate of transformation uncertainty, and incorporate that estimate in decisions about feature pairings.

Another class of recognition methods are those that search the space of possible pairings without invoking a transformation estimate. Essentially, these methods achieve a match by identifying a subisomorphism between two graphs—one representing the model and the other representing the image—in which nodes and arcs denote features and their relations. Attributes associated with nodes and arcs record geometric measurements, such as the position of one feature with respect to another, and geometric uncertainty is represented by uncertainty in those attributes. (Interpretation tree methods [8] are equivalent in that they search a space of pairings while observing constraints among small groups of features.)

In the PREMIO system by Camps, Shapiro and Haralick [6], a 2D characteristic view is modeled by a graph with nodes denoting line segments and arcs denoting junctions and groups of junctions. By defining a cost function for graph matches and an associated admissable heuristic, they are able to match model and image graphs using heuristic rather than exhaustive search. The model includes Gaussian distributions characterizing how many nodes and relations are expected to match, and how much attributes are expected to differ from their norms. All features of one view, however, share common distributions.

Burns and Riseman [5] describe a contrasting approach where a graph, called a view description network, models a hierachy of components from low-level, generic primitives, through high-level, object-specific arrangements, to entire views of an object. Matching proceeds in stages from low level to high. Again, attributes are characterized by distributions to accommodate varation in appearance.

Our previous work [11] used a graph-matching method resembling both PREMIO and view description networks. Like PREMIO, we used a cost function and heuristic search to match graphs. Like Burns and Riseman, our graphs represented a range of features, from simple to complex, and each relation was characterized by a distribution to accommodate variation in appearance. The present work retains these aspects of our earlier work.

Graph and alignment methods each emphasize different types of constraints in the matching process. Graph matching respects the topological and geometrical relations among small groups of features, ensuring, for example, that model line segments sharing a common junction are paired with image line segments sharing a similar junction. Alignment, on the other hand, seeks to ensure that feature pairings are all consistent with respect to some single viewpoint hypothesis. The present work combines these two approaches so that both topological and viewpoint constraints direct the search. It will be shown that this has advantages over using only one type of constraint or the other.

A third class of methods are those that search transformation space. These include the generalized Hough transform and geometric hashing, which both involve the accumulation of votes in an array of bins that tesselates a transformation space or parameter space. Early methods let the tesselation granularity determine the permissible mismatch between model and image features, giving no control over the uncertainty of individual features. Such control may come from using weighted votes (e.g., Rigoutsos and Hummel [14], although they assume the same uncertainty for all features) or indexing functions learned from examples [2].

Breuel [4] and Cass [7] have reported methods that avoid tesselation and instead subdivide transformation space recursively to localize arbitrary regions of it. Their algorithms achieve excellent performance by applying constraints that, in transformation space, are of a particularly simple form. That form is ensured by using a bounded-error model of uncertainty whereby a model feature may match an image feature anywhere within some $\epsilon$ distance of the model feature's projected position in the image. However, in a system that learns models from positive training examples only, there is no principled way to determine an appropriate error bound for a model feature (just as there is no way to determine an upper bound for human ages by studying a subpopulation). Moreover, for some image features there is empirical evidence that errors in localizing the features are better modeled by Gaussian distributions than by bounded ones [15]. The method described in this paper uses multivariate Gaussian distributions to represent the expected positions of model features.

## 3 Method

The graph representations used for images and models, and the algebraic representation used for viewpoint transformations, are described first in sections 3.1 and 3.2. A match, consisting of a set of feature pairings and an estimate of the viewpoint transformation, is evaluated by a match quality measure, which is then described (section 3.3). Matching seeks to maximize this measure. One component of it is an estimate of the probability that two features match, given their respective position distributions and an estimate of the viewpoint transformation; this component is described in section 3.4 while other components have been described in previous work [11]. Section 3.5 describes how a viewpoint transformation is estimated from a set of feature pairings, and section

3.6 ties these pieces together in describing the procedure for finding a match. That matching procedure is used both for recognition, and as part of a procedure described in section 3.7 for learning models from training images.

### 3.1 Image and model representations

An image input to the system, whether for training or for recognition, is represented by an *image graph*. Nodes of this graph denote features detected in the image while arcs denote abstraction and composition relations among features. A feature may, for example, be a segment of intensity edge, a particular arrangement of such segments, the response of a corner detector, or a region of uniform color. A typical image will be described by numerous features of various types, scales, and degrees of abstraction, some found by low-level detectors and others found by grouping processes.

Formally, an image graph $G$ is denoted by a tuple $\langle F, R\rangle$, where $F$ is a set of tokens denoting image features and $R$ is a relation over elements of $F$. An image feature token $f_k \in F$ is a tuple $\langle t_k, \mathbf{a}_k, \mathbf{b}_k, \mathbf{C}_k\rangle$ where $t_k$ is the feature's type, $\mathbf{a}_k$ is a vector of attributes describing the feature, $\mathbf{b}_k$ is its measured position, and $\mathbf{C}_k$ describes the uncertainty in that position. Attributes are numeric measurements of the feature's intrinsic properties, such as its curvature if it is a circular arc or its interior angle if it is a junction. Feature position and its uncertainty are described below. Finally, a relation in $R$ is a tuple $\langle k, l_1, \ldots, l_m\rangle$, indicating that image feature $k$ was found by grouping or abstracting image features $l_1$ through $l_m$.

A model describes a characteristic view of an object. Like an image, it is represented by a graph with nodes denoting features and arcs denoting abstraction and composition relations among them. However, a model graph also includes information to support estimates of the probability that a model feature will be observed, and the probability that it will have particular attributes when observed. This information is accumulated from the training images used to generate the model.

Formally, a model graph $\overline{G}$ is denoted by a tuple $\langle \overline{F}, \overline{R}, \overline{m}\rangle$, where $\overline{F}$ is a set of tokens denoting model features, $\overline{R}$ is a relation over elements of $\overline{F}$, and $\overline{m}$ is the number of training images used to produce $\overline{G}$. A model feature token $\overline{f_j} \in \overline{F}$ is a tuple $\langle \overline{t_j}, \overline{m}_j, \overline{A}_j, \overline{B}_j\rangle$, where $\overline{t_j}$ is the feature's type, $\overline{m}_j$ is the number of training images in which the feature was observed, and $\overline{A}_j$ and $\overline{B}_j$ are the sequences of attribute vectors and positions drawn from those training images. Finally, a relation in $\overline{R}$ is a tuple $\langle j, l_1, \ldots, l_m\rangle$, indicating that model feature $j$ is a grouping or abstraction of model features $l_1$ through $l_m$.

### 3.2 Coordinate systems

A feature's position is specified by a location, orientation, and scale expressed in terms of a 2D, Cartesian coordinate system. Image features are located in an *image coordinate system* identified with pixel rows and columns. Model features are located in a *model coordinate system* that is arbitrarily fixed and used for all features within a model graph.

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

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

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

The $xy\theta s$ scheme is the more convenient for measuring feature position while the $xyuv$ scheme, as we shall see, is convenient for estimating viewpoint from feature pairings. The two schemes are related by $\theta = tan^{-1}(v/u)$ and $s = (u^2 + v^2)^{\frac{1}{2}}$. Where it is not otherwise clear, we will indicate a scheme with the superscripts ${}^{xy\theta a}$ and ${}^{xyuv}$.

Viewpoint is represented by a *viewpoint transformation*, which is a 2D similarity transformation bringing paired image and model features into close correspondence. The $xyuv$ scheme allows such a transformation to be expressed as a linear operation with the advantage that it can then be estimated from a set of feature pairings by solving a system of linear equations.[1]

We take the viewpoint transformation, $T$, to be from image to model coordinates, using it to transform the position of an image feature before comparing it with that of a model feature. A transformation consisting of a rotation by $\theta_t$, a scaling by $s_t$, and a translation by $[x_t\ y_t]$ (in that order), can be expressed in two ways as a linear operation. We use both. In one case, the position being transformed is represented by a matrix, $\mathbf{A}_k$:

$$\mathbf{b}'_k = \begin{bmatrix} x'_k \\ y'_k \\ u'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_k & -y_k \\ 0 & 1 & y_k & x_k \\ 0 & 0 & u_k & -v_k \\ 0 & 0 & v_k & u_k \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \end{bmatrix} = \mathbf{A}_k \mathbf{b}_t. \tag{1}$$

[1] Ayache and Faugeras [1], among others, have also used this formulation to render the transformation as a linear operation.

In the other case, the position being transformed is represented by a vector, $\mathbf{b}_k$:

$$\mathbf{b}'_k = \begin{bmatrix} x'_k \\ y'_k \\ u'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} u_t & -v_t & 0 & 0 \\ v_t & u_t & 0 & 0 \\ 0 & 0 & u_t & -v_t \\ 0 & 0 & v_t & u_t \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ u_k \\ v_k \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ 0 \\ 0 \end{bmatrix} = \mathbf{A}_t \mathbf{b}_k + \mathbf{x}_t. \tag{2}$$

The result of applying $T$ to the position $\mathbf{b}_k$ is denoted $T(\mathbf{b}_k)$.

### 3.3 Match quality measure

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. Together, the pairings and transformation are called a *match*. The match should be a "good" one that jointly maximizes both the number of features paired and the quality or closeness of those pairings. Described here is a match quality measure that makes this notion precise and leads to a procedure for finding near-optimal matches.

The match quality measure extends that reported in [11] to include an evaluation of how well the viewpoint transformation brings features into correspondence.[2] A set of pairings is represented by the tuple $E = \langle e_1, e_2, \ldots \rangle$, where $e_j = k$ if model feature $j$ matches image feature $k$, and $e_j = \perp$ if it matches nothing. The hypothesis that the object is present in the image is denoted by $H$. Match quality is associated with the probability that this hypothesis is correct given a set of pairings and a viewpoint transformation. Bayes theorem allows us to write this probability as:

$$\mathrm{P}(H \mid E, T) = \frac{\mathrm{P}(E \mid T, H)\ \mathrm{P}(T \mid H)}{\mathrm{P}(E \wedge T)} \mathrm{P}(H). \tag{3}$$

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

$$\mathrm{P}(H \mid E, T) \approx \prod_j \frac{\mathrm{P}(e_j \mid T, H)}{\mathrm{P}(e_j)} \frac{\mathrm{P}(T \mid H)}{\mathrm{P}(T)} \mathrm{P}(H). \tag{4}$$

We assume that all viewpoint transformations are equally likely a priori, and thus $\mathrm{P}(T)$ is a constant. $\mathrm{P}(H)$ is the prior probability that the modeled object is present in the image; it can be

[2] There are also minor differences in notation.

estimated from the proportion of training images that matched the model and were used to create it. $\mathrm{P}(T \mid H)$ can be taken as identical to $\mathrm{P}(T)$, or it can be estimated from past training images by keeping some record of the transformations found upon matching the model to those images.

In what follows, the random outcome $\tilde{e}_j = k$ denotes 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 $j$ failed to match an image feature during training. We use a Bayesian estimator, a uniform prior, and the $\bar{m}$ and $\bar{m}_j$ statistics defined in section 3.1:

$$\mathrm{P}(\tilde{e}_j = \perp \mid T, H) = 1 - \mathrm{P}(\tilde{e}_j \neq \perp \mid T, H) \approx 1 - \frac{\bar{m}_j + 1}{\bar{m} + 2} \tag{5}$$

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:

$$\begin{aligned} \mathrm{P}(\tilde{e}_j = k \mid T, H) \quad \approx \quad & \mathrm{P}(\tilde{e}_j \neq \perp \mid T, H)\, \mathrm{P}(\tilde{\mathbf{a}}_j = \mathbf{a}_k \mid \tilde{e}_j \neq \perp, H) \\ & \mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H). \end{aligned} \tag{6}$$

The $\mathrm{P}(\tilde{e}_j \neq \perp)$ term is estimated as shown in equation 5. The $\mathrm{P}(\tilde{\mathbf{a}}_j = \mathbf{a}_k)$ term is estimated using the series of attribute vectors $A_j$ recorded with model feature $j$, and a non-parametric density estimator described in [11]. Estimation of the $\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k))$ term (the probability that model feature $j$ will match an image feature at position $\mathbf{b}_k$ with viewpoint transformation $T$) is described below, in section 3.4.[3]

Estimates of the prior probabilities are based, in part, on measurements from a large, random collection of images typical of those in which the object will be sought. From this *milieu* collection we obtain prior probabilities of encountering various types of features with various

[3] For simplicity, our notation ignores the difference between probability masses and probability densities. $\mathrm{P}(\tilde{e}_j)$ is a mass because $\tilde{e}_j$ assumes discrete values, whereas $\mathrm{P}(\tilde{\mathbf{a}}_j)$ and $\mathrm{P}(\tilde{\mathbf{b}}_j)$ are densities because $\tilde{\mathbf{a}}_j$ and $\tilde{\mathbf{b}}_j$ are continuous. But since equation 4 divides each conditional probability mass by a prior probability mass, and each conditional probability density by a prior probability density, we can safely omit the distinction between masses and densities in this context.

values of attribute vectors. The prior probability of a feature occuring at any particular position is estimated by assuming that features are uniformly distributed throughout a bounded region of the model coordinate system.

The match quality measure is equated not with $\mathrm{P}(H \mid E, T)$ itself, but with its logarithm so that exponentials (as in the Gaussian of equation 11, below) are eliminated and multiplications are replaced by additions. With constants also eliminated, the match quality measure becomes:

$$g(E,T) = \log \mathrm{P}(H) + \sum_j \left( \log \mathrm{P}(e_j \mid T, H) - \log \mathrm{P}(e_j) \right) . \tag{7}$$

### 3.4 Estimating feature match probability

The probability that a model feature is matched by an image feature depends, in part, on the positions of the two features and on the viewpoint transformation that brings them into alignment. This position- and transformation-dependent portion of the feature match probability is represented by the $\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H)$ term in equation 6. To estimate that probability, we transform the image feature's position into model coordinates according to the viewpoint transformation and compare it with the model position (see figure 1). Both positions as well as the viewpoint transformation are characterized by Gaussian probability density functions (pdfs) so that the comparison takes into account their respective uncertainties.

Image feature $k$'s position is conveniently characterized by a Gaussian pdf in $xy\theta s$ image coordinates, with mean $\mathbf{b}_k^{xy\theta s}$ and covariance matrix $\mathbf{C}_k^{xu\theta s}$. The mean is the feature's position as measured in the image. Because our system's feature detectors and grouping processes do not supply uncertainty estimates, we define the covariance matrix using system parameters $\sigma_l$, $\sigma_\theta$, and $\sigma_s$, which are our estimates of the standard deviations in measurements of location, orientation, and scale. Moreover, since the orientation of a large feature can usually be measured more accurately than that of a small feature, the feature's scale is considered when estimating its orientation uncertainty. The covariance matrix we use is

$$\mathbf{C}_k^{xy\theta s} = \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 \\ 0 & 0 & (\frac{\sigma_\theta}{s_k})^2 & 0 \\ 0 & 0 & 0 & \sigma_s^2 \end{bmatrix}$$

Before transforming a feature's position from image coordinates to model coordinates the position is expressed in $xyuv$ image coordinates so that equation 1 or 2 can be used to apply the

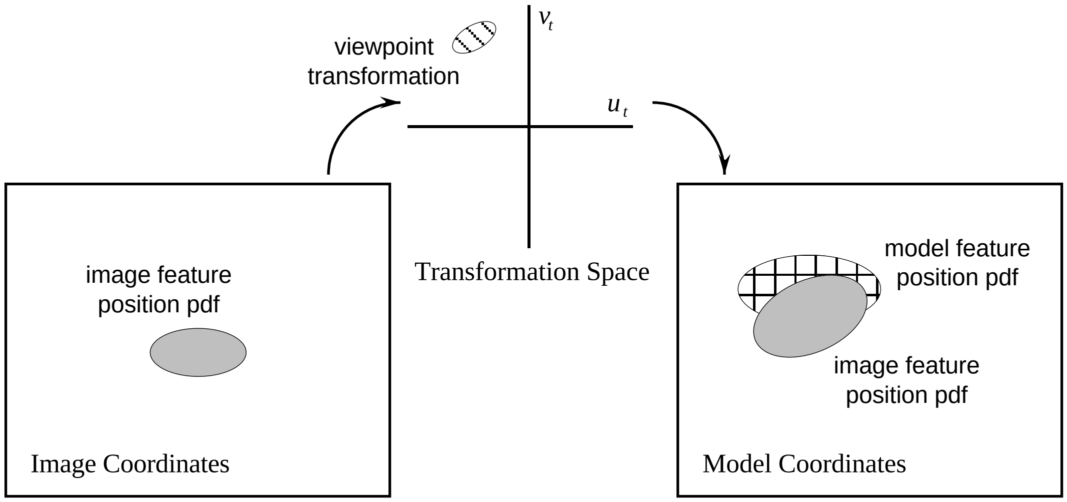


Figure 1: An image feature's position is transformed from image coordinates (left) to model coordinates (right) according to an estimate of the viewpoint transformation (center), while a model feature's position is estimated in model coordinates (right). Uncertainty in the positions and the transformation are all characterized by Gaussian distributions. Overlap of the two distributions in model coordinates corresponds to the probability that the two features match given the viewpoint transformation and their respective positions.

transformation. A pdf that is Gaussian in $xy\theta s$ coordinates is not necessarily Gaussian in $xyuv$ coordinates. Nevertheless a good approximating Gaussian can be obtained in $xyuv$ coordinates if, as in this case, the $\theta$ and $s$ covariances are not large. The approximation places the $xyuv$ mean at the same position as the $xy\theta s$ mean, and aligns the Gaussian envelope radially, away from the $[u\ v]$ origin (see figure 2). Its mean and covariance matrix are

$$\begin{aligned}
\mathbf{b}_k^{xyuv} &= [x_k \;\; y_k \;\; s_k \cos\theta_k \;\; s_k \sin\theta_k] \text{ and} \\
\mathbf{C}_k^{xyuv} &= \mathbf{R} \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 \\ 0 & 0 & \sigma_s^2 & 0 \\ 0 & 0 & 0 & \sigma_\theta^2 \end{bmatrix} \mathbf{R}^{\mathrm{T}}, \\
\text{where } \mathbf{R} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\theta_k & -\sin\theta_k \\ 0 & 0 & \sin\theta_k & \cos\theta_k \end{bmatrix}.
\end{aligned}$$

The viewpoint transformation is characterized by a Gaussian pdf over $[x_t\ y_t\ u_t\ v_t]$ vectors. The

Figure 2: The Gaussian distribution of an image feature's position in $xy\theta s$ coordinates (left) is approximated by a Gaussian distribution in $xyuv$ coordinates (right), with the parameters of the approximating distribution determined as shown.

mean and covariance of this pdf, $\mathbf{t}$ and $\mathbf{C}_t$, are obtained from the estimator described below, in section 3.5.

We now use the viewpoint transformation to transform the image feature's position from $xy\theta s$ to $xyuv$ model coordinates. If we disregard the uncertainty in the transformation estimate, we will obtain a Gaussian pdf in model coordinates with mean $\mathbf{A}_k\mathbf{t}$ and covariance $\mathbf{A}_t\mathbf{C}_k\mathbf{A}_t^{\mathrm{T}}$. On the other hand, if we disregard the uncertainty in the image feature position, we will obtain a Gaussian pdf in model coordinates with mean $\mathbf{A}_k\mathbf{t}$ and covariance $\mathbf{A}_k\mathbf{C}_t\mathbf{A}_k^{\mathrm{T}}$. But when both the image feature's position and the transformation are characterized by Gaussian pdfs, the pdf in model coordinates cannot be characterized as Gaussian. At best we can approximate it with a Gaussian pdf, and this we do using the mean and covariance given by

$$\mathbf{b}_{kt} = \mathbf{A}_k\mathbf{t} \text{ and} \tag{8}$$

$$\mathbf{C}_{kt} \approx \mathbf{A}_t\mathbf{C}_k\mathbf{A}_t^{\mathrm{T}} + \mathbf{A}_k\mathbf{C}_t\mathbf{A}_k^{\mathrm{T}}. \tag{9}$$

The position of the model feature, $j$, is also characterized by a Gaussian pdf in $xyuv$ model coordinates. Its mean $\mathbf{b}_j$ and covariance $\mathbf{C}_j$ are simply estimated from the series of position vectors, $\bar{B}_j$, that the model records for that feature.[4]

We can now estimate the probability that $j$ matches $k$ according to their position pdfs and the

[4] Two practical considerations enter into the estimation of $\mathbf{C}_j$. First, when $\bar{B}_j$ contains too few samples for a reliable estimate of $\mathbf{C}_j$, the estimate that $\bar{B}_j$ yields is blended with another determined by system parameters. Second, minimum variances are imposed on $\mathbf{C}_j$ to overcome situations where $\bar{B}_j$ has zero variance in some dimension.

transformation pdf. This is done by integrating, over all positions $\mathbf{r}$, the probability that both the image feature is at $\mathbf{r}$ and the model feature matches something at $\mathbf{r}$:

$$\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H) \approx \int_{\mathbf{r}} \mathrm{P}(\tilde{\mathbf{r}}_j = \mathbf{r})\, \mathrm{P}(\tilde{\mathbf{r}}_{kt} = \mathbf{r})\, \mathrm{d}\mathbf{r}, \tag{10}$$

Here $\mathbf{r}$ ranges over $xyuv$ model coordinates while $\tilde{\mathbf{r}}_j$ and $\tilde{\mathbf{r}}_{kt}$ are random variables drawn from the Gaussian distributions $\mathrm{N}(\mathbf{b}_j, \mathbf{C}_j)$ and $\mathrm{N}(\mathbf{b}_{kt}, \mathbf{C}_{kt})$. That the integral is a Gaussian in $\mathbf{b}_j - \mathbf{b}_{kt}$ can be seen from the fact that it is essentially a convolution of two Gaussians. Indeed, it is equivalent to

$$\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H) \approx G(\mathbf{b}_j - \mathbf{b}_{kt}, \mathbf{C}_j + \mathbf{C}_{kt}) \tag{11}$$

where $G(\mathbf{x}, \mathbf{C})$ is a Gaussian with zero mean and covariance $\mathbf{C}$. Equations 8, 9, and 11 give us our desired estimate.

### 3.5 Estimating viewpoint transformation

From a series of feature pairings we wish to estimate a viewpoint transformation that will maximize our match quality measure. Fortunately, the transformation is applied as a linear operation (equation 1) and the match quality measure effectively sums the squares of the distances between paired model and image features (equations 7 and 11). Consequently, this is a linear least-squares estimation problem for which good algorithms exist.

The estimation problem is formulated as follows. Each pairing $\langle j, k \rangle$ of model and image features is related by the transformation $\mathbf{t}$ and a residual error $\tilde{\mathbf{e}}$:

$$\mathbf{A}_k\, \mathbf{t} = \mathbf{b}_j + \tilde{\mathbf{e}}. \tag{12}$$

Here $\mathbf{A}_k$ is the matrix representation of image feature $k$'s mean position, $\mathbf{t}$ is the transformation estimate represented by the vector $[x_t\; y_t\; u_t\; v_t]$, and $\mathbf{b}_j$ is the vector representation of model feature $j$'s mean position. The residual $\tilde{\mathbf{e}}$ is assumed to have a Gaussian distribution whose covariance, $\mathbf{C}_j$, can be estimated from the series of position vectors, $\overline{B}_j$, recorded by the model. Through a process known as "whitening", we can rewrite this relation so that the residual has unit variance. Here $\mathbf{U}_j$ denotes the upper triangular square root of $\mathbf{C}_j$ (i.e., $\mathbf{C}_j = \mathbf{U}_j \mathbf{U}_j^{\mathrm{T}}$):

$$\mathbf{U}_j^{-1}\, \mathbf{A}_k\, \mathbf{t} = \mathbf{U}_j^{-1}\, \mathbf{b}_j + \tilde{\mathbf{e}}',\ \text{where } \tilde{\mathbf{e}}' \sim \mathrm{N}(0, I).$$

A series of feature pairings gives us a series of such relations. From those, a linear least-squares estimator determines both the transformation $\mathbf{t}$ that minimizes the sum of the residual errors, and its covariance $\mathbf{C}_t$.

During a match search based on iterative alignment, feature pairings are adopted sequentially. We need to refine the transformation estimate with each new pairing or group of pairings adopted so that an improved estimate can then be used to identify additional pairings. Thus a recursive estimator is used.

The square root information filter (SRIF) is a recursive estimator that is particularly well suited for this problem. Compared to the Kalman filter it is numerically more stable, it is faster for batched measurements, and it has the nice property of computing the total residual error as a side effect [3]. As its name implies, 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 $\langle j, k \rangle$ of model and image features:

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

Then, 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 estimates of the viewpoint transformation and its covariance are needed, they can be obtained by

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

This requires only back substitution since $\mathbf{R}_i$ is triangular. The SRIF also makes the total residual error available as $\mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}$, which conveniently corresponds to the $\log \mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H)$ component of our match quality measure. Thus, following each update of the transformation estimate, the match quality measure for the new transformation can be computed easily, without the need to re-evaluate equation 11 with the new transformation and all previous feature pairings.

### 3.6 Matching procedure

In matching model and image graphs for the purpose of recognition, we seek a match $\langle E, T\rangle$ that maximizes the match quality measure (equation 7). Although it does not appear possible to find the optimal match through anything less than exhaustive search, in practice near-optimal matches can be found quickly by iterative alignment. Here we describe how iterative alignment is performed in our case.

Alignment starts from hypothesized feature pairings that each provide an initial estimate of the viewpoint transformation. To choose these hypotheses, all possible pairings of higher-level features are ranked according to the contribution each would make to the match quality measure. The rank of the pairing $\langle j, k\rangle$ is given by:

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

This ranking favors pairings where the model feature has a high likelihood of matching, the two features have similar attribute values, and the resulting transformation estimate's variance is small. Moreover, because the component of $\mathrm{P}(\tilde{e}_j = k \mid T, H)$ that depends on $T$ is a Gaussian, its maximum over $T$ can be computed readily. A search is begun from each of the several highest-ranked pairings.

From an initial pairing, the search proceeds by identifying additional consistent pairings, adopting the best of them, and using those to update the transformation estimate. Again, possible pairings are ranked according to the contribution each will make to the match quality measure:

$$g_j(k; E, T) = \begin{cases} \log \mathrm{P}(\tilde{e}_j = k \mid T, H) - \log \mathrm{P}(\tilde{e}_i = k) & \text{if } \langle j, k\rangle \text{ is consistent with } E \\ 0 & \text{otherwise} \end{cases}$$

This ranking favors the same criteria as the ranking of initial pairings (equation 13), while further requiring that pairings be consistent with those already adopted and favoring pairings whose feature positions correspond closely according to the transformation estimate. Possible pairings are placed on a priority queue so that, once all pairings have been evaluated, the queue contains a few dozen of the best. Then, if any queued pairings can be considered ambiguous because they conflict with other queued pairings, those ambiguous pairings are downrated so that they will be postponed in favor of less ambiguous ones. Finally, the highest-ranked pairings are adopted and used to update the transformation estimate.

Backtracking is performed when ambiguity forces a choice among conflicting pairings, and a search branch is terminated when no additional pairings can be identified to improve the match quality measure. From several starting hypotheses and the various search branches that result from backtracking we obtain a number of consistent matches. As matches are found, only the best match is retained, and its match quality measure provides a threshold for pruning subsequent search branches.

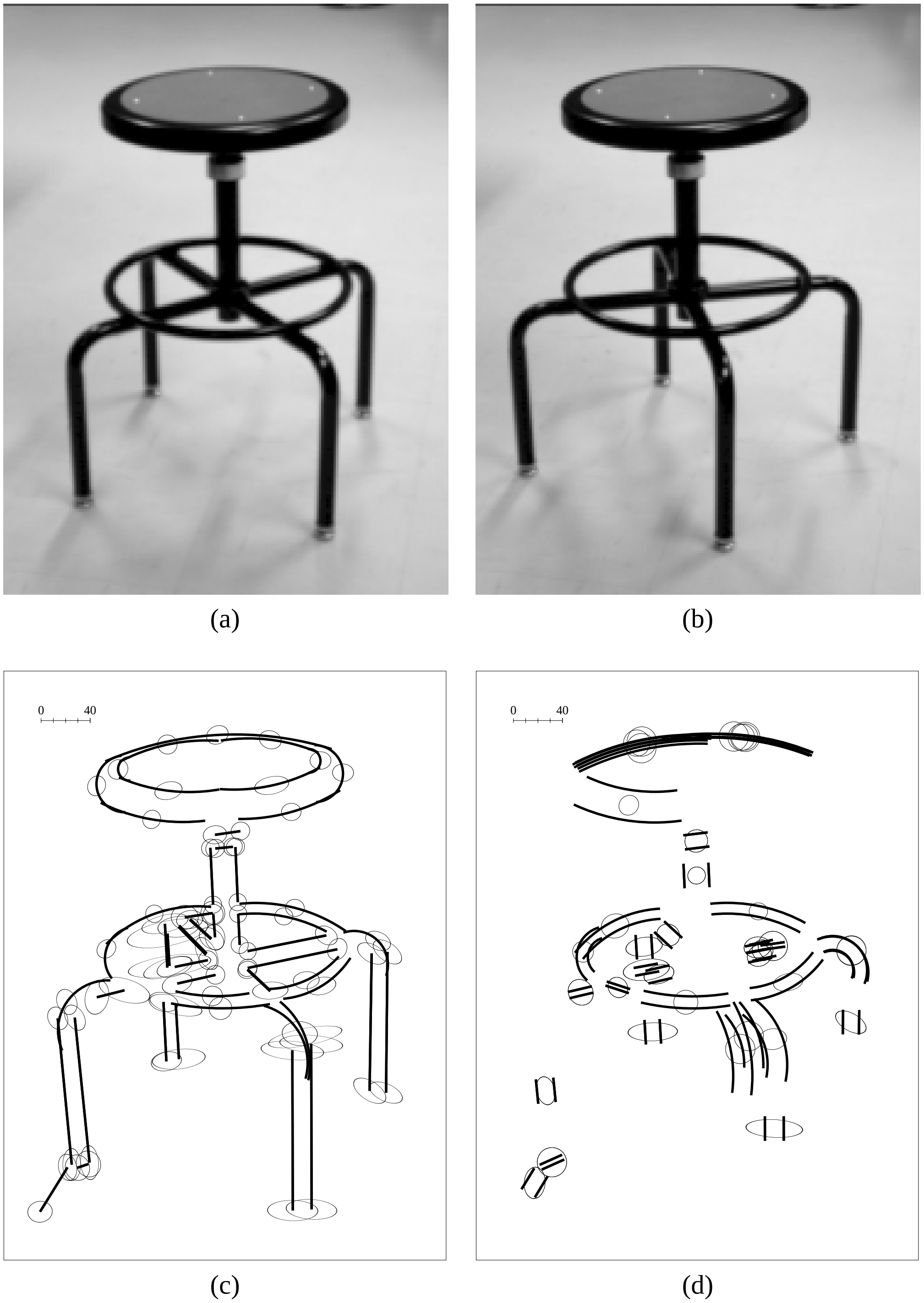
### 3.7 Model learning procedure

Since the model learning procedure has been described elsewhere [11] we will only summarize it here. An initial model graph is formed from the first training image's graph. The 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 $\overline{A}_j$ and $\overline{B}_j$. Some 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.

## 4 Experimental results

The method has been implemented using facilities of the Vista computer vision environment [13]. The system recognizes 3D objects in 2D intensity images, employing a repertoire of features chosen for describing the appearance of manufactured objects. Straight and circular segments of intensity edges are the lowest-level features. These are augmented by features representing various perceptually-significant groupings, including junctions, pairs and triples of junctions, pairs of parallel segments, chains of such pairs, and convex regions. Features that are rotationally symmetric, such as straight lines, are simply represented by multiple tokens, one per orientation.

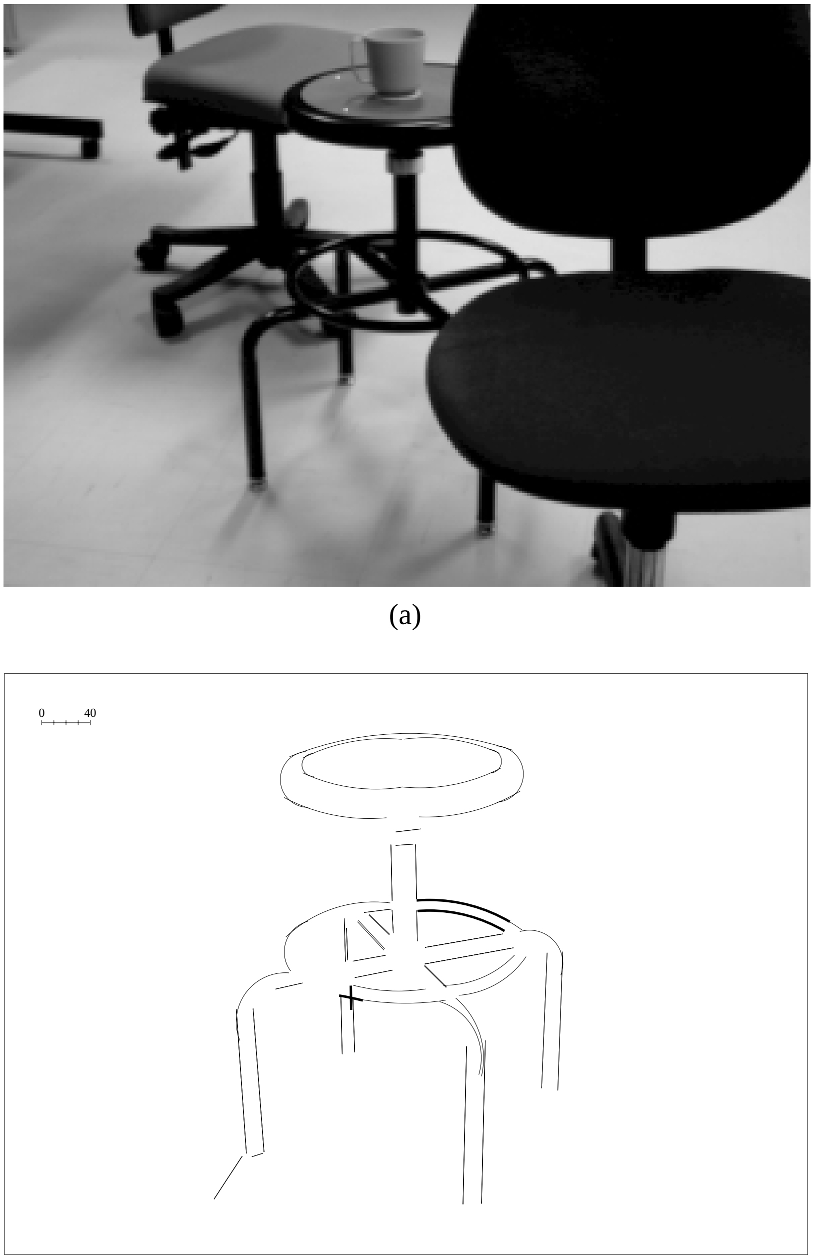
Presented here are two examples that illustrate some aspects of the method. Figure 3 shows a model of a characteristic view of a stool learned from nine training images acquired over a 20-degree range of viewpoint. Figure 4 shows that model being used to recognize the stool in a test image. As evident from the model depiction and from figure 5, features of the model differ widely in positional uncertainty. Some differences are due to shifts in the relative positions of



(a) (b)

(c) (d)

Figure 3: Nine training images spanning 20 degrees of viewing angle, from (a) to (b), yield a single characteristic view model. Among model features, those denoting straight and circular segments of intensity edge are shown in (c); those denoting pairs of parallel segments are shown in (d). Ellipses depict two standard deviations of feature location uncertainty.



(a)

(b)

Figure 4: A cluttered test image (a) in which the partially-occluded stool is recognized (b). This match begins with a pairing of junctions, shown in bold, that is rated highly by equation 13 primarily due to the image feature's intrinsic attributes. Matching proceeds with a pairing of parallel arcs, also shown in bold, that is favored in part due to its model feature's low positional uncertainty (apparent in figure 3(d)). Model features representing segments of intensity edge are shown as light lines projected into the image according to the final estimate of the viewpoint transformation.

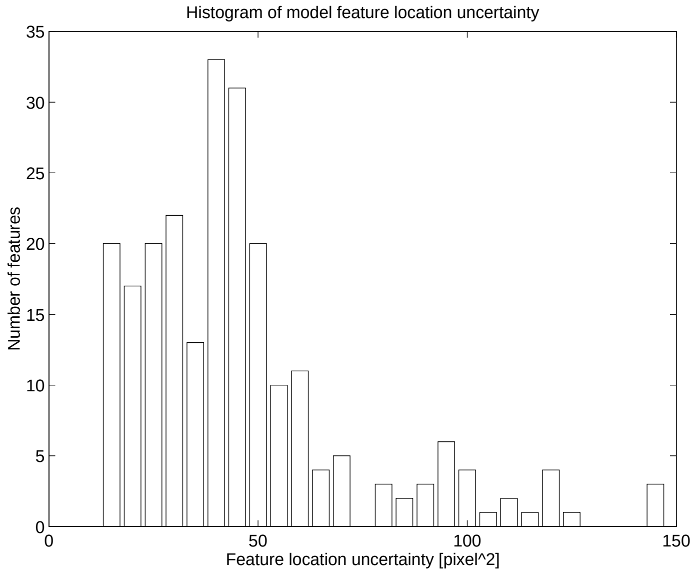


Figure 5: Features of the stool model vary widely in positional uncertainty, as shown by this histogram of feature location uncertainty. Here, location uncertainty is measured as the area of a one-standard-deviation ellipse about the model feature's expected location.

features with changing viewpoint: the seat and post remain fixed, for example, while the legs shift in various directions. Others are due to inherent differences in the accuracy of localizing various types of features: for example, a right-angle junction might be better localized than an oblique or acute one. Differences would be even greater for a flexible object.

A second example of model learning is shown in figure 6. After eight training images the model includes a few features whose positions are quite uncertain due to lighting effects; any pairings involving these features will have little influence on the viewpoint transformation estimate and the match quality measure.

This example also illustrates how both topological and geometric relations among features contribute matching constraints. Along the leftmost edge of the object, parallel line segments are so closely spaced that their position distributions largely overlap; position alone provides little help in choosing the correct pairings for these features. However, the line segments are also components of more distinctive features, including junctions and a parallel pair. Pairings for these other features are less ambiguous. Once those pairings are adopted, they constrain pairings involving the line segments through topological relations represented as arcs in the model and image graphs.

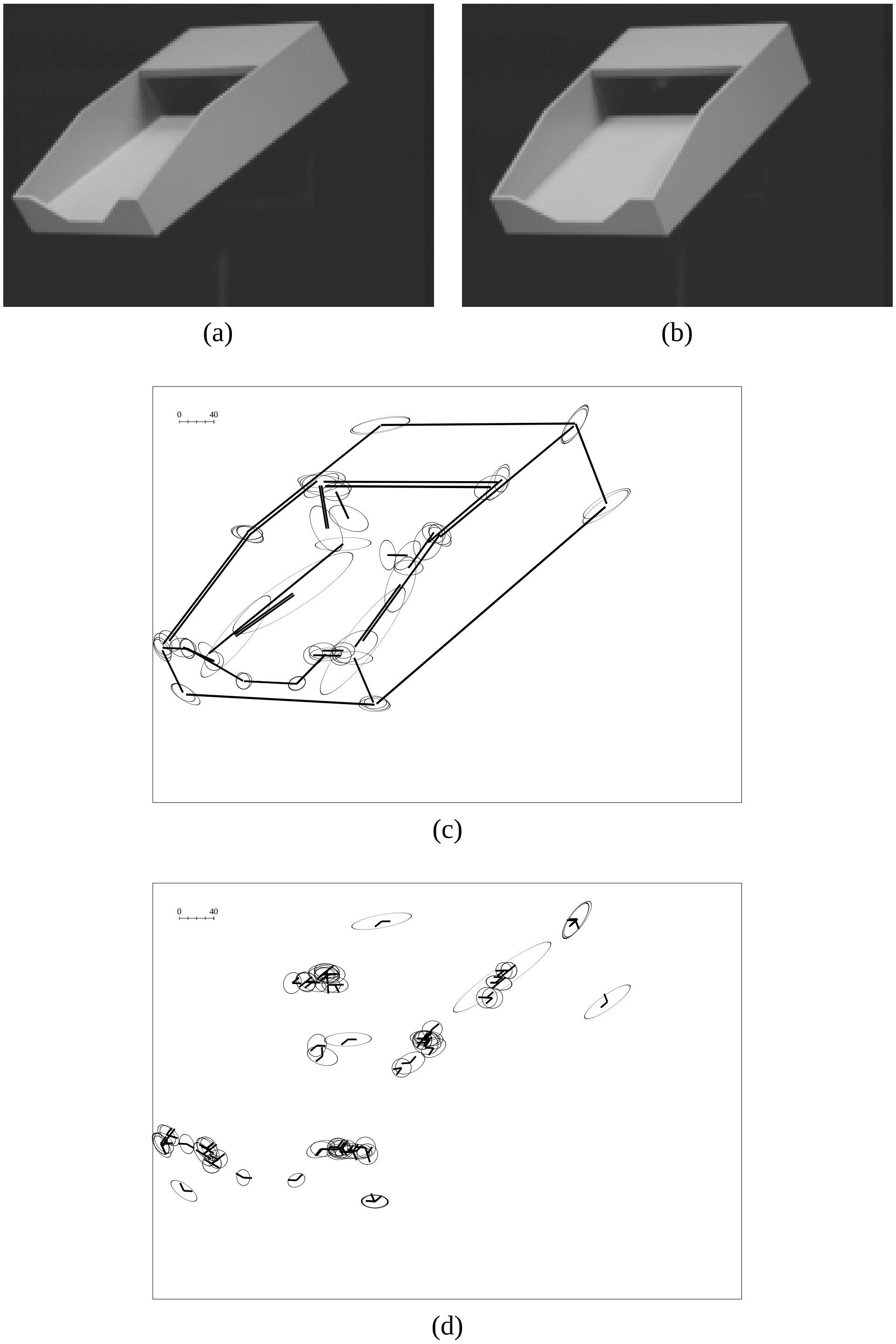


Figure 6: Eight training images spanning 14 degrees of viewing angle, from (a) to (b), yield a single characteristic view model. Among model features, those denoting straight segments of intensity edge are shown in (c); those denoting junctions are shown in (d). Ellipses depict two standard deviations of feature location uncertainty. Those that are exaggerated correspond to unreliable features.

## 5 Discussion

We have described an object recognition method that uses four characterizations of each model feature:

- A characterization of the feature's diagnostic power, representing how likely it is to be detected and how strongly it indicates the presence of the object.
- A characterization of the intrinsic attributes of matching image features.
- A characterization of the positions of matching image features.
- A set of abstraction and composition relations involving other model features, as represented by the topology of the model graph.

All four are acquired from training images and used opportunistically to constrain the search for a match between a model and an image. The search typically begins by matching a feature that is very specific due to its intrinsic attributes, very likely to be matched in any image of the object, and otherwise rare; usually this will be a high-level feature representing a grouping of several simpler ones. Remaining matches are then constrained not only by their intrinsic attributes, but also by their positions with respect to an estimated viewpoint and by topological relations relative to the features already matched. In some cases position alone can identify unambiguous matches, while in other cases topological relations effectively choose among nearby features in an image.

In comparison, alignment methods and methods that search transformation space have generally used only feature positions, and not information about features' topological relations. Methods based on graph matching have generally not used feature positions directly, but considered only topological relations and the relative positions of small groups of features.

Three of these four characterizations (all but graph topology) are represented probabilistically, allowing a model to represent not just one canonical appearance of an object but rather an expected range of appearance variation. This allows a single model to represent the appearance of a 3D object over a range of similar viewpoints, a flexible object in a variety of similar configurations, or an entire class of similar but distinctive objects.

This paper has focused on how feature positions, in particular, can be represented probabilistically and used in matching. When the viewpoint transformation is restricted to a 2D similarity transformation, $xyuv$ coordinate systems allow that transformation to be represented as a linear operation. Moreover, when uncertainties in the transformation and model feature positions are modeled by Gaussian distributions, an optimal transformation can be estimated from feature pairings by a linear least-squares estimator.

It should be noted that the least-squares formulation can consider the uncertainty in image feature positions or model feature positions, but not both. Considering uncertainty in both model and image feature positions requires a total least-squares formulation, which is not so readily solved.

We have chosen to consider uncertainty in model feature positions, which in terms of equation 12 means that errors in $\mathbf{b}_k$ and $\mathbf{t}$, but not $\mathbf{A}_k$, are considered in the least-squares solution. Instead one could choose to consider uncertainty in image feature positions, as Ayache and Faugeras [1] have done, by reversing the approach and solving for a model-to-image viewpoint transformation. However, the pdfs of image feature positions carry considerably less information than those of model features, which become highly individualized during model acquisition. Our choice allows the viewpoint transformation to be constrained most by model features whose positions vary little, and less by those whose positions vary greatly.

The explicit representation of uncertainty allows a model to describe appearance more completely and accurately over a range of viewing conditions. We believe that, as a consequence, fewer 2D characteristic views will be needed to describe a 3D object for a given level of recognition performance. In experiments reported here, we have achieved good performance with 2D characteristic views spanning 14 to 20 degrees of viewpoint azimuth, indicating that fewer than one hundred views may be needed to model appearance from all viewing directions.

This paper has described a representation for 2D characteristic view models, a method called *probabilistic alignment* for matching such models to images, and a procedure for learning the models from training images. Experiments now in progress will better characterize the performance of the probabilistic alignment method and the nature of the probability distributions acquired through learning. In future work we plan to add a conceptual clustering procedure that subdivides training images among clusters corresponding to distinct characteristic views [12]. Together, these components will form a system capable of automatically learning to recognize a 3D object from any viewpoint.

## References

[1] N. Ayache and O. D. Faugeras, “HYPER: A new approach for the recognition and

positioning of two-dimensional objects," *IEEE T-PAMI* **PAMI-8** (1986), pp. 44-54.

[2] J. S. Beis and D. G. Lowe, "Learning indexing functions for 3-D model-based object recognition," In *Proc. CVPR* (1994), pp. 275-280.

[3] G. J. Bierman, *Factorization methods for discrete sequential estimation*, New York : Academic Press (1977).

[4] T. M. Breuel, "Fast recognition using adaptive subdivisions of transformation space," In *Proc. CVPR* (1992), pp. 445-451.

[5] J. B. Burns and E. M. Riseman, "Matching complex images to multiple 3D objects using view description networks," In *Proc. CVPR* (1992), pp. 328-334.

[6] O. I. Camps, L. G. Shapiro, and R. M. Haralick, "Object recognition using prediction and probabilistic matching," In *Proc. of the IEEE/RSJ International Conf. on Intelligent Robots and Systems* (July 1992), pp. 1044-1052.

[7] T. A. Cass, "Polynomial-time object recognition in the presence of clutter, occlusion, and uncertainty," In *Proc. ECCV*, (1992), pp. 834-842.

[8] W. E. L. Grimson, *Object Recognition by Computer: The Role of Geometric Constraints*, MIT Press (1990).

[9] D. P. Huttenlocher and S. Ullman "Recognizing solid objects by alignment with an image," *Int. J. Comput. Vision* **5** (1990), pp. 195-212.

[10] D. G. Lowe, "The viewpoint consistency constraint," *Int. J. Comput. Vision* **1** (1987), pp. 57-72.

[11] A. R. Pope and D. G. Lowe, "Learning object recognition models from images," In *Proc. ICCV* (1993), pp. 296-301.

[12] A. R. Pope and D. G. Lowe, "Learning 3D object recognition models from 2D images," In *Proc. AAAI Fall Workshop on Machine Learning in Computer Vision* (1993).

[13] A. R. Pope and D. G. Lowe, "Vista: A software environment for computer vision research," In *Proc. CVPR* (1994), pp. 768-772.

[14] I. Rigoutsos and R. Hummel, "Distributed Bayesian object recognition," In *Proc. CVPR* (1993), pp. 180-186.

[15] W. M. Wells III, *Statistical Object Recognition*, Ph.D. thesis, MIT Dept. of Electrical Engineering and Computer Science (1992).