Learning Appearance Models for Object Recognition

Arthur R. Pope[1] and David G. Lowe[2]

[1] David Sarnoff Research Center, CN 5300, Princeton, NJ 08543–5300
[2] Dept. of Computer Science, University of British Columbia, Vancouver, B.C., Canada V6T 1Z4

Abstract. We describe how to model the appearance of an object using multiple views, learn such a model from training images, and recognize objects with it. The model uses probability distributions to characterize the significance, position, and intrinsic measurements of various discrete features of appearance; it also describes topological relations among features. The features and their distributions are learned from training images depicting the modeled object. A matching procedure, combining qualities of both alignment and graph subisomorphism methods, uses feature uncertainty information recorded by the model to guide the search for a match between model and image. Experiments show the method capable of learning to recognize complex objects in cluttered images, acquiring models that represent those objects using relatively few views.

1 Introduction

The multiple-view object recognition approach models an object with a series of views, each describing the object's appearance over a small range of viewing conditions. Systems adopting this approach generally assume that all features of a model view have the same likelihood of being detected and the same positional uncertainty, perhaps because better models are difficult to obtain [8, p. 182]. Clearly, though, features differ in incidence, localization accuracy, and stability. A system that learns models from example images can directly measure these differences. This paper describes how to represent feature uncertainty in a multiple-view model, learn such models from training images, and recognize objects with them.

Information about feature uncertainty 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 best localized can contribute most to an estimate of the object's position; and features whose positions vary most can be sought over the largest image neighborhoods. Our matching method, based on both iterative alignment and graph matching, achieves these goals. 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, with the goal that there be as many groups as necessary, but no more. 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.

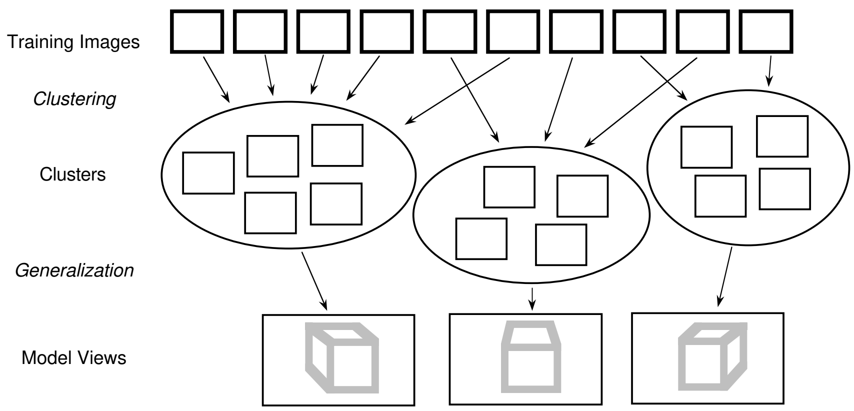


Fig. 1. Learning a multiple-view model from training images requires a clustering of the training images and a generalization of each cluster's contents.

2 Related Research

2.1 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 [1] and 2D–3D matching [12]. 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 [12]. 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 [22, p. 5]. 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.

Some recognition methods are based on matching attributed graphs in which nodes and arcs represent features and their relations, and attributes record measurements. PREMIO [6] uses Gaussian distributions characterizing the expected number of feature and relation matches, and the expected deviation of attributes from their norms, to define a graph similarity measure that guides a fast, heuristic search for matches; all features of one model view, however, share common distributions. In view description networks [5], attribute distributions are determined by regularly sampling idealized features.

Whereas graph matching enforces topological and geometric relations among groups of features—e.g., ensuring that model line segments sharing a common junction are paired with image line segments sharing a similar junction—alignment enforces the viewpoint consistency constraint [13]. By combining these two approaches, we gain advantages from employing all constraints.

Recognition methods that search transformation space by accumulating votes may use feature uncertainty to weight votes (e.g., [17], although they assume the same uncertainty for all features). Methods that avoid tesselating the space [4, 7] have required the use of bounded error models of feature uncertainty to achieve their high efficiency. However, in our situation, where models are learned from positive training examples only, there is no way to determine error bounds; we use Gaussian error models instead. Empirical evidence [23, ch. 3] supports this choice, at least for some features.

2.2 Learning Appearance Models

Some approaches model an object as a subspace within a large space of possible appearances, and use principal components analysis to obtain a concise description of the particular subspace occupied by a given set of training examples (e.g., [21, 14]). However, applications of this approach have used global appearance representations, such as entire images, and thus they have not supported recognition of occluded objects.

Connell and Brady [9] have described a system that learns an appearance model of a 2-D object (or one view of a 3-D object), using structures of localized features. The system incorporates many interesting ideas. They use graphs to represent the part/whole and adjacency relations among object regions described by smoothed local symmetries (ribbon shapes). An attribute of a region, such as its elongation or curvature, is encoded symbolically by the presence or absence of additional graph nodes according to a Gray code. A structural learning procedure forms a model graph from multiple example graphs, most commonly by deleting any nodes not shared by all graphs (the well-known dropping rule for generalization). Similarity between two graphs is measured by a purely syntactic measure: simply by counting the nodes they share. Consequently, this system accords equal importance to all features, and it uses a somewhat arbitrary metric for comparing attribute values.

Learning a multiple-view model from real images requires some means of comparing and clustering appearances. Although several researchers have clustered images rendered from CAD models and thus avoided the feature correspondence problem, only a few have clustered real images. Among them, Gros [11] measures the similarity of an image pair as the proportion of matching shape features, whereas Seibert and Waxman [20] use a vector clustering algorithm with fixed-length vectors encoding global appearance. Our method, in comparison, uses a clustering measure based on objective performance criteria (accuracy and efficiency), and an appearance representation less affected by occlusion.

3 Method

Representations used for images, models, and transformations are described in Sects. 3.1 and 3.2. A match, comprising a set of feature pairings and an aligning transformation, is rated by the measure described in 3.3. One component of this measure estimates the probability that two features match given their respective position distributions and an aligning transformation; it is described in 3.4; other components have been described previously [15]. The method of estimating a transformation from feature pairings is described in 3.6. A matching procedure, described in 3.5, uses the match quality measure and transformation estimator to match model features with image features.

The matching procedure is used both to learn a model from training images and to recognize a modeled object in a scene. The learning procedure is described in 3.7. Recognition combines the matching procedure with an indexing procedure for selecting likely model views from a model database, and a verification procedure for deciding whether a match presents sufficient evidence that an object is present. Suitable indexing and verification methods have been described elsewhere (e.g., [2, 19]), and will not be discussed here.

A more complete description of the entire approach may be found in [16].

3.1 Image and Model Representations

An image is represented by a graph with nodes denoting features and arcs denoting abstraction and composition relations among them. 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 is described by many features of various types, scales, and degrees of abstraction, some found by low-level detectors, others by grouping.

Formally, an image graph G is a tuple $\langle F, R\rangle$ where F is a set of image features and R is a relation over elements of F. A feature $f_k \in F$ is a tuple $\langle t_k, \mathbf{a}_k, \mathbf{b}_k, \mathbf{C}_k\rangle$; t_k is the feature's type, $\mathbf{b}_k$ and $\mathbf{C}_k$ are the mean and covariance of its image position, and $\mathbf{a}_k$ is a vector of descriptive attributes (e.g., the curvature of a circular arc, the interior angle of a junction). An element of R, $\langle k, l_1, \ldots, l_n \rangle$, indicates that feature k groups or abstracts features l_1 through l_n.

An object is modeled by a series of model views. A model view is represented by a graph similar to an image graph, but one that includes information for estimating the probability that a feature will be found in various positions and with various attributes. It describes, for each model feature, a distribution of where that feature may be expected to be found once the model and image have been satisfactorily aligned by a transformation.

Formally, a model graph $\bar{G}$ is a tuple $\langle \bar{F}, \bar{R}, \bar{m} \rangle$, where $\bar{F}$ is a set of model features, $\bar{R}$ is a relation over elements of $\bar{F}$, and $\bar{m}$ is the number of training images used to produce $\bar{G}$. A feature $\bar{f}_j \in \bar{F}$ is a tuple $\langle \bar{t}_j, \bar{m}_j, \bar{A}_j, \bar{B}_j \rangle$; $\bar{t}_j$ is the feature's type, $\bar{m}_j$ is the number of training images in which the feature was observed, and $\bar{A}_j$ and $\bar{B}_j$ are the sequences of attribute vectors and positions drawn from those training images. The mean and covariance matrix of $\bar{B}_j$ are denoted $\mathbf{b}_j$ and $\mathbf{C}_j$. $\bar{R}$ is defined similarly to R.

3.2 Coordinate Systems

Feature positions are specified by 2D 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 schemes are used:

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

We will prefer the $xy\theta s$ scheme for measuring feature positions and the $xyuv$ scheme 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 necessary, superscripts $^{xy\theta s}$ and ^{xyuv} indicate which scheme is in use.

A 2D similarity transformation T is used to align features.[3] Fortunately, the $xyuv$ scheme allows T to be estimated from feature pairings by solving a system of linear equations.[4] The transformation of image position $\mathbf{b}_k = [x_k\ y_k\ u_k\ v_k]$ involving a rotation by θ_t, a scaling by s_t, and a translation by $[x_t\ y_t]$ (in that order), has two linear formulations, both used here:

$$T(\mathbf{b}_k) = \begin{bmatrix} 1 & 0 & x_k & -y_k \\ 0 & 1 & y_k & x_k \\ 0 & 0 & u_k & -v_k \\ 0 & 0 & v_k & u_k \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \end{bmatrix} = \mathbf{A}_k \mathbf{b}_t \text{ and}$$

[3] There is an analogous formulation using affine transformations with advantages only in modeling 3D planar objects.

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

$$T(\mathbf{b}_k) = \begin{bmatrix} u_t & -v_t & 0 & 0 \\ v_t & u_t & 0 & 0 \\ 0 & 0 & u_t & -v_t \\ 0 & 0 & v_t & u_t \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ u_k \\ v_k \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ 0 \\ 0 \end{bmatrix} = \mathbf{A}_t \mathbf{b}_k + \mathbf{x}_t \ .$$

3.3 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. Our match quality measure quantifying these goals extends that reported in [15] to include an evaluation of how well the transformation aligns 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 E and T, which Bayes' theorem lets us write as

$$\mathrm{P}(H \mid E, T) = \frac{\mathrm{P}(E \mid T, H)\, \mathrm{P}(T \mid H)}{\mathrm{P}(E \wedge T)} \mathrm{P}(H) \ . \tag{1}$$

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:

$$\mathrm{P}(H \mid E, T) \approx \prod_j \frac{\mathrm{P}(e_j \mid T, H)}{\mathrm{P}(e_j)} \frac{\mathrm{P}(T \mid H)}{\mathrm{P}(T)} \mathrm{P}(H) \ . \tag{2}$$

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:

$$\mathrm{P}(\tilde{e}_j = \perp \mid T, H) = 1 - \mathrm{P}(\tilde{e}_j \neq \perp \mid T, H) \approx 1 - \frac{\bar{m}_j + 1}{\bar{m} + 2} \ . \tag{3}$$

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:

$$\begin{aligned} \mathrm{P}(\tilde{e}_j = k \mid T, H) \approx \mathrm{P}(\tilde{e}_j \neq \perp \mid T, H)\, &\mathrm{P}(\tilde{\mathbf{a}}_j = \mathbf{a}_k \mid \tilde{e}_j \neq \perp, H) \\ &\mathrm{P}(\tilde{\mathbf{b}}_j = T(\mathbf{b}_k) \mid \tilde{e}_j \neq \perp, T, H) \ . \end{aligned} \tag{4}$$

$\mathrm{P}(\tilde{e}_j \neq \perp)$ is estimated as in (3). $\mathrm{P}(\tilde{\mathbf{a}}_j = \mathbf{a}_k)$ is estimated using the series of attribute vectors $\bar{A}_j$ recorded with model feature j, and a non-parametric density estimator described in [15]. 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. 3.4.[5]

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.

3.4 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 (4). 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

[5] For simplicity, our notation does not distinguish probability mass and probability density. $\mathrm{P}(\tilde{e}_j)$ 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 (2) divides each conditional probability mass by a prior probability mass, and each conditional probability density by a prior probability density, here we can safely neglect the distinction.

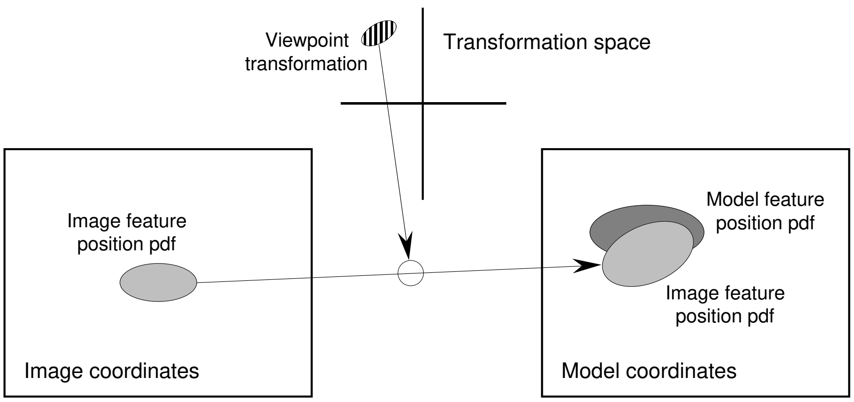


Fig. 2. An aligning transformation maps an image feature's position to model coordinates, where it is compared with a model feature's position to yield one component of the probability that the image and model features match.

covariance matrix $\mathbf{C}_k^{xyuv}$ that aligns the Gaussian envelope radially, away from the $[u\ v]$ origin:

$$\mathbf{b}_k^{xyuv} = [x_k \ \ y_k \ \ s_k \cos\theta_k \ \ s_k \sin\theta_k] \text{ and}$$

$$\mathbf{C}_k^{xyuv} = \mathbf{R} \begin{bmatrix} \sigma_l^2 & 0 & 0 & 0 \\ 0 & \sigma_l^2 & 0 & 0 \\ 0 & 0 & \sigma_s^2 & 0 \\ 0 & 0 & 0 & \sigma_\theta^2 \end{bmatrix} \mathbf{R}^{\mathrm{T}} \ ,$$

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

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. 3.6. 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

$$\mathbf{b}_{kt}^{xyuv} = \mathbf{A}_k\mathbf{t} \text{ and}$$

$$\mathbf{C}_{kt}^{xyuv} = \mathbf{A}_t \mathbf{C}_k^{xyuv} \mathbf{A}_t^{\mathrm{T}} + \mathbf{A}_k \mathbf{C}_t \mathbf{A}_k^{\mathrm{T}} \ .$$

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.[6]

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 $\mathbf{r}$:

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

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:

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

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.

3.5 Matching Procedure

Matches between a model graph and an image graph are identified by a process that combines iterative alignment and graph matching. First, possible pairings of higher-level features are ranked according to the contribution each would make to the match quality measure. The pairing $\langle j, k \rangle$ receives the rating

$$g_j(k) = \max_T \log \mathrm{P}(\tilde{e}_j = k \mid T, H) - \log \mathrm{P}(\tilde{e}_j = k) \ , \tag{5}$$

favoring pairings where j has a high likelihood of matching, j and k have similar attributes, 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 the highest-ranked pairings. Each estimates 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. Consistency is judged with respect to previously adopted pairings and the relations recorded by graph arcs. Again, a pairing is rated according to its match quality measure contribution:

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

[6] 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. Also, minimum variances are imposed on $\mathbf{C}_j$ to overcome situations where $\bar{B}_j$ has zero variance in some dimension.

This favors the same qualities as (5), while also favoring pairings aligned closely by the estimated transformation. To postpone ambiguous choices, highly-ranked but conflicting pairings of a common feature are downgraded.

As alignments yield matches, the best is retained and its match quality measure provides a threshold for cutting off subsequent alignments. Alignments are attempted until a match is found meeting acceptance criteria (e.g., a minimum fraction of edges matched) or resource limits are reached.

3.6 Estimating 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) [3] 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

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

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:

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

3.7 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 (Fig. 1).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.

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 (e.g., COBWEB [10]), 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 (or match) the training images.

The minimum description length principle [18] 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 those that can be encoded concisely, and it favors accurate models as those that allow the 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. 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. 3.5) 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 $\bar{G}_i$:

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

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.

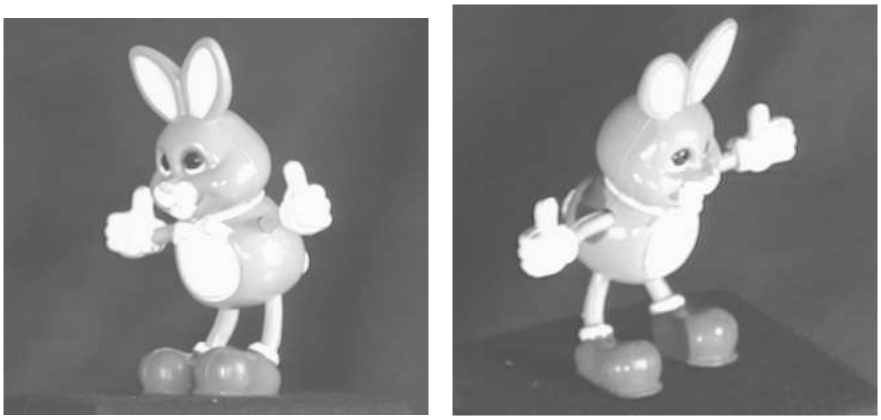
Generalizing Training Images to Form a Model Graph. Within each cluster, training images are merged to form a single model graph that represents a generalization of those images. An initial model graph is formed from the first training image's graph. That model graph is then matched with each subsequent 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 $\bar{B}_j$. 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 in a system that recognizes 3–D objects in 2–D intensity images using a basic repertoire of features. The lowest-level features are straight, circular and elliptical edge segments. Additional features, representing perceptually-significant groupings, are junctions, pairs and triples of junctions, pairs of parallel segments, and convex regions. Although this feature repertoire has proven adequate for recognizing a wide variety of objects, it can also be readily extended to extend the range of objects handled by the system.

Figs. 3 through 6 present one example of model learning and recognition. Other examples of objects the system has learned to recognize are shown in Fig. 7.

The learning procedure was applied to 112 training images of a bunny acquired at 5° intervals over camera elevations of 0° to 25° and azimuths of 0° to 30° (Fig. 3). The learning procedure clustered these training images to produce the groups shown in Fig. 4. (Although presenting training images to the system in different orders yielded different clusterings, those clusterings were all qualitatively similar in that each contained approximately the same number of groups and the same distribution of group sizes.) Each group of training images was generalized to produce a model graph describing the range of appearances contained in that group. One such model graph is depicted in Fig. 5; other model graphs describe other views of the bunny in similar detail. Note that individual model features differ widely in uncertainty, as indicated by the standard deviation ellipses shown in Fig. 5.



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

Fig. 3. Two of 112 training images used to learn an appearance model of the bunny.

Fig. 6 shows an image in which the bunny is successfully recognized by matching features of the image with those of one of the model graphs. In this case, the model graph that best matches the image is that derived from group D of the training images, which is as expected since group D encompasses that aspect of the bunny visible in the scene.

5 Summary

We have presented a general method for recognizing complex, real-world objects using appearance models acquired from training images.

Appearance in an image is represented by an attributed graph of discrete features and their relations, with a typical object described by many features. Since one object can vary greatly in appearance when viewed under different conditions, a model is represented by a probability distribution over such graphs. The range of this distribution is divided among characteristic views, allowing a simplified representation for each view as a model graph of independent features.

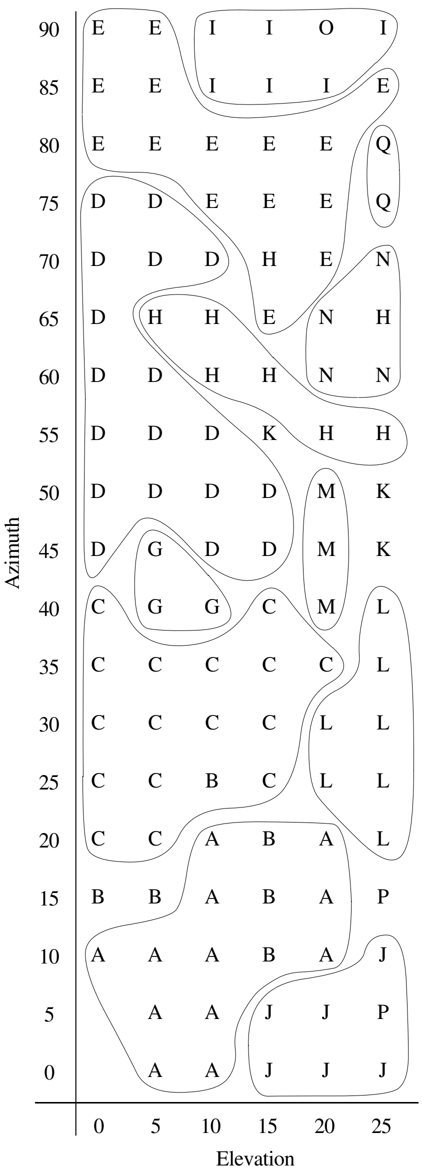


Fig. 4. Seventeen groups, designated A through Q, were formed from the 112 bunny training images. Contours delineate the approximate scope of the model views defined by some of the groups. Note, however, that because the model views are defined probabilistically, their boundaries are actually indefinite.

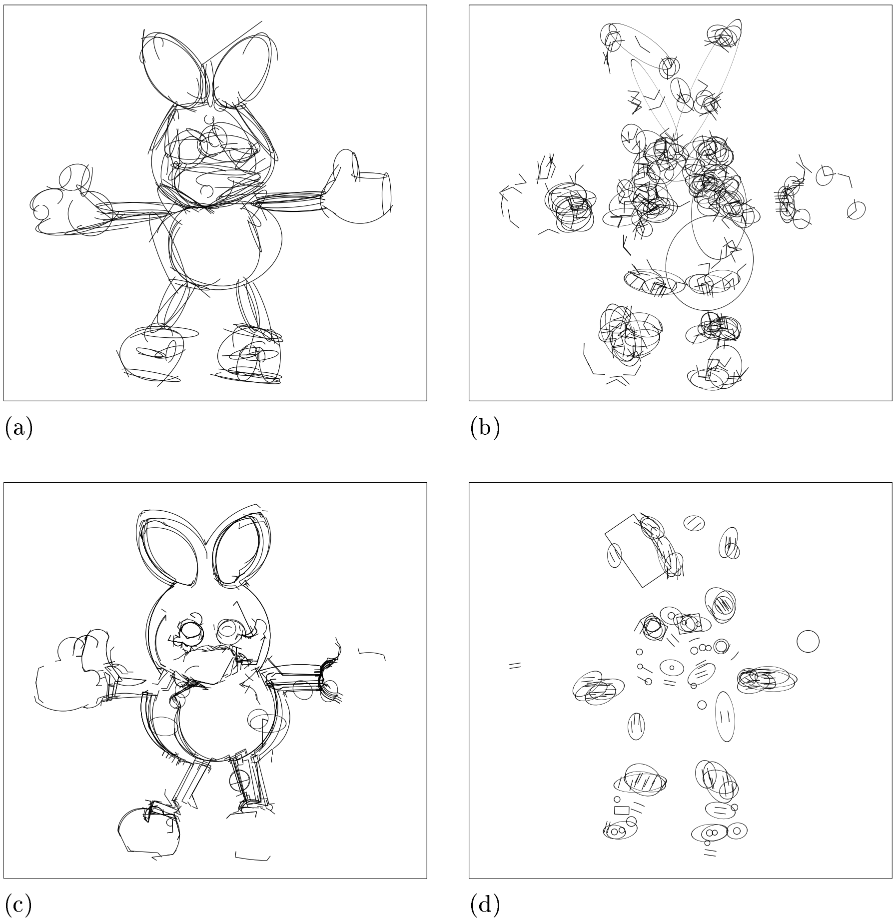


Fig. 5. Shown here are selected features of the model graph obtained by generalizing the training images assigned to group D (delineated in Fig. 4). Each feature is drawn at its mean location. (a) Features denoting edge segments. (b) Features denoting edge segment junctions. (c) Groups of edge segment junctions. (d) Groups denoting parallel pairs and closed regions of edge segments. In (b) through (d), ellipses showing 2 s.d.'s of feature location uncertainty are drawn for those features found in a majority of training images.

A model feature is described by probability distributions for probabilities of detection, various internal attribute values, and various image positions. All three distributions are estimated from samples supplied by training images.

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 the distributions recorded by the model. The measure leads naturally to an efficient matching procedure called probabilistic alignment. In searching for

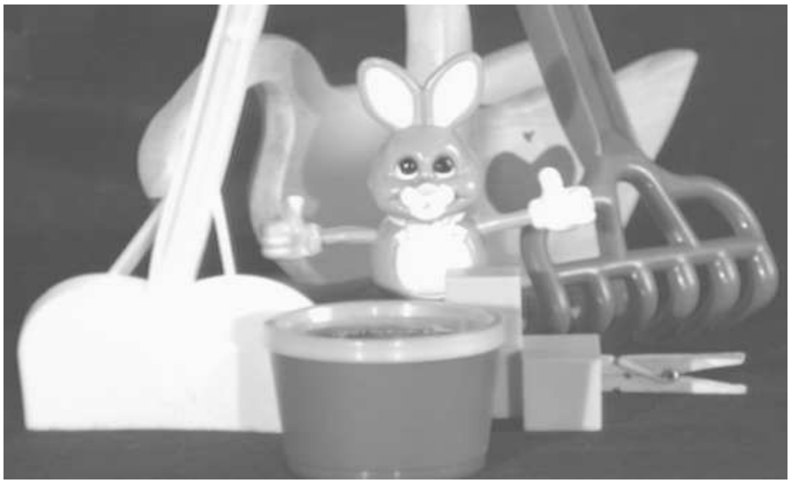


Fig. 6. Bunny test image. *Above:* Image. *Below:* Edge segment features of the image that were matched by those of the model. Additional features, such as junctions and regions, were also matched, but they are not shown here.

a solution, the procedure can employ constraints arising both from the topology of the model graph and from the probability distributions describing individual features.

The model learning procedure has two components. A conceptual clustering component identifies groups of training images that correspond to characteristic views by maximizing a global measure of clustering quality. That measure uses the minimum description length principle to combine a simplicity criterion favoring concise models, with a fit criterion based on the match quality measure. A generalizing component merges the images within each group to form a model graph representing a generalization of that group. It uses the matching procedure to determine correspondences among the group's images.

In principle the method can recognize any object by its appearance, given a sufficient range of training images, sufficient storage for model views, and an appropriate repertoire of features. In practice, however, highly flexible objects will require impractical numbers of training images and model views. For such objects, reducing the complexity of models, learning and recognition remains a topic for further study.

References

1. N. Ayache, O.D. Faugeras. HYPER: A new approach for the recognition and positioning of two-dimensional objects. *IEEE Trans. Patt. Anal. Mach. Intell.* PAMI–8:44–54, 1986.
2. J.S. Beis, D.G. Lowe. Learning indexing functions for 3-D model-based object recognition. In *Proc. Conf. Computer Vision and Patt. Recognit.*, 275–280, 1994.
3. G.J. Bierman. *Factorization Methods for Discrete Sequential Estimation*. Academic Press, 1977.
4. T.M. Breuel. Fast recognition using adaptive subdivisions of transformation space. In *Proc. Conf. Computer Vision and Patt. Recognit.*, 445–451, 1992.
5. J.B. Burns, E.M. Riseman. Matching complex images to multiple 3D objects using view description networks. In *Proc. Conf. Computer Vision and Patt. Recognit.*, 328–334, 1992.
6. O.I. Camps, L.G. Shapiro, R.M. Haralick. Object recognition using prediction and probabilistic matching. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Systems*, 1044–1052, July 1992.
7. T.A. Cass. Polynomial-time object recognition in the presence of clutter, occlusion, and uncertainty. In *Proc. European Conf. on Computer Vision*, 834–842, 1992.
8. C.H. Chen, P.G. Mulgaonkar. Automatic vision programming. *CVGIP: Image Understanding* 55:170–183, 1992.
9. J.H. Connell, M. Brady. Generating and generalizing models of visual objects. *Artificial Intell.* 31:159–183, 1987.
10. D.H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning* 2:139–172, 1987.
11. P. Gros. Matching and clustering: Two steps towards automatic object model generation in computer vision. In *Proc. AAAI Fall Symp.: Machine Learning in Computer Vision*, AAAI Press, 1993.
12. Y. Hel-Or, M. Werman. Pose estimation by fusing noisy data of different dimensions. *IEEE Trans. Patt. Anal. Mach. Intell.* 17:195–201, 1995.
13. D.G. Lowe. The viewpoint consistency constraint. *Int. J. Computer Vision* 1:57–72, 1987.
14. H. Murase, S.K. Nayar. Learning and recognition of 3-D objects from brightness images. In *Proc. AAAI Fall Symp.: Machine Learning in Computer Vision*, 25–29, AAAI Press, 1993.
15. A.R. Pope, D.G. Lowe. Learning object recognition models from images. In *Proc. Int. Conf. Computer Vision*, 296–301, 1993.

16. A.R. Pope. *Learning to Recognize Objects in Images: Acquiring and Using Probabilistic Models of Appearance.* Ph.D. thesis, Univ. of British Columbia, 1995. WWW `http://www.cs.ubc.ca/spider/pope/Thesis.html`.
17. I. Rigoutsos, R. Hummel. Distributed Bayesian object recognition. In *Proc. Conf. Computer Vision and Patt. Recognit.*, 180–186, 1993.
18. J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics* 11:416–431, 1983.
19. K.B. Sarachik, W.E.L. Grimson. Gaussian error models for object recognition. In *Proc. Conf. Computer Vision and Patt. Recognit.*, 400–406, 1993.
20. M. Seibert, A.M. Waxman. Adaptive 3-D object recognition from multiple views. *IEEE Trans. Patt. Anal. Mach. Intell.* 14:107–124, 1992.
21. M.A. Turk, A.P. Pentland. Face recognition using eigenfaces. In *Proc. Conf. Computer Vision and Patt. Recognit.*, 586–591, 1991.
22. S. van Huffel, J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis.* SIAM, 1991.
23. W.M. Wells III. *Statistical Object Recognition.* Ph.D. thesis, MIT, 1992.

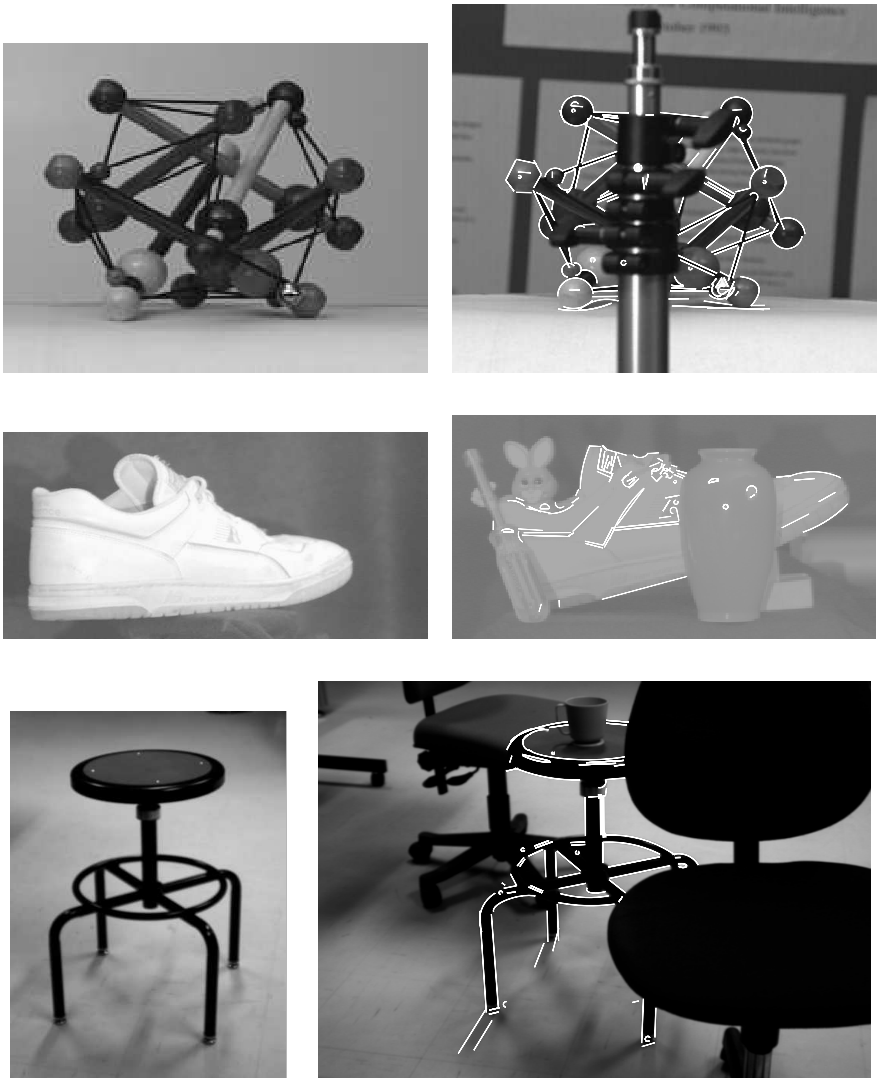


Fig. 7. Other examples of objects the system has learned to recognize. *Left*: One element drawn from each object's set of training images. *Right*: Recognition of the objects in test images.