Probabilistic Learning of Three-Dimensional Object Models*

Gregory Provan (PROVAN@CAMIS.STANFORD.EDU)
Pat Langley (LANGLEY@CS.STANFORD.EDU)
Institute for the Study of Learning and Expertise
2164 Staunton Court, Palo Alto, CA 94306 USA

Thomas O. Binford (BINFORD@CS.STANFORD.EDU)
Robotics Laboratory, Computer Science Department
Stanford University, Stanford, CA 94305 USA

Abstract

In this paper we report on an approach to learning object models for use in recognition and reconstruction. Our framework represents objects in an image using generalized cylinders and organizes knowledge about classes of objects in a Bayesian network. The recognition process involves propagating evidence through this inference network, whereas learning relies on updating of the network's conditional probabilities based on training cases. We report preliminary experimental results with synthetic data that suggest our method improves its recognition accuracy with experience. We also consider our framework's relation to other research on learning object knowledge for image understanding.

1. Introduction

The image-understanding process relies on accurate knowledge. This statement holds for all levels of visual processing, but seems especially true at the later stages, where object recognition and reconstruction require models of objects or object classes that occur in the domain. And though one could, in principle, enter such domain-specific knowledge manually, this technique can be expensive, time consuming, and a source of errors. Most existing image-understanding systems include just a few models in their libraries, due to the difficulty of entering models and organizing them in memory. For domains with hundreds or thousands of object classes, some automated process of model construction seems to be necessary.

In this paper we present one approach to automating the acquisition and revision of object models that draws on recent work in machine learning. In particular, we consider representational, performance, and induction methods that have two properties useful for computer vision: the ability to operate over object descriptions at multiple levels of aggregation; and the use of probabilities to handle the uncertainty inherent in image understanding. We also report experimental evidence, using synthetic but realistic data, that the learning algorithm leads to more accurate recognition of object classes as it gains experience in a domain, and that it can take advantage of approximate domain knowledge when present.

Although our effort holds some features in common with other work on learning object models, it differs in the central role played by two representational assumptions which we will discuss shortly: that three-dimensional objects can be usefully represented as generalized cylinders and that models of object classes can be usefully encoded in Bayesian networks. As we will see, the performance and learning algorithms follow directly from these assumptions.

2. Representing Objects in Images

Before one can recognize objects in an image, one must first be able to represent those objects. The literature on computer vision contains many responses to this basic issue. Some researchers describe objects in terms of low-level features (e.g., Murase & Nayar, 1993; Pope & Lowe, 1993). Others represent objects using a set of characteristic views that describe the objects' appearance from alternative perspectives (e.g., Dickinson, Pentland, & Rosenfeld, 1992). We prefer three-dimensional over two-dimensional representations because the latter are subject to much more variation across different perspectives. For example, characteristic views require one to store many distinct 'object' descriptions in memory for each physical object, which leads to high costs in the match process. This greater complexity should also require more training cases during learning, since the large number of views means there are many more parameters to determine, compared to the perspective-invariant models used by 3D schemes.

* This research was was supported by Grant No. N00014-94-1-0746 from the Office of Naval Research, with partial funding from the Advanced Research Projects Agency.

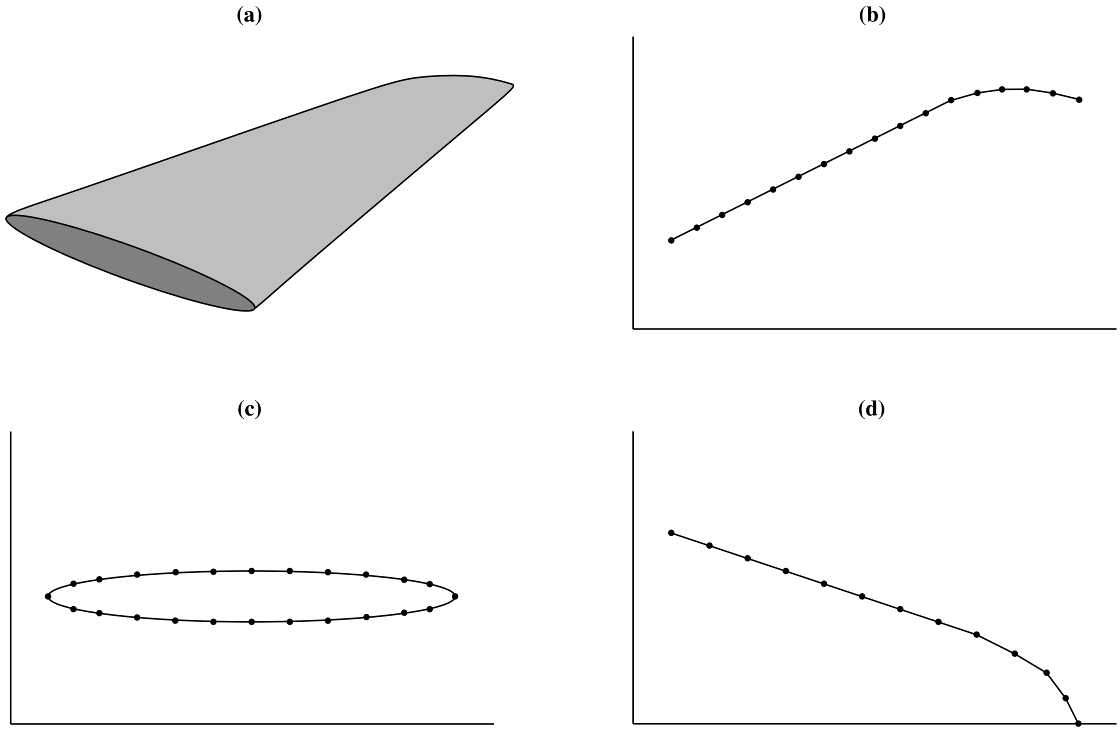

Figure 1: (a) The wing of an airplane, along with the (b) axis function, (c) cross section, and (d) sweep function that characterize it as a generalized cylinder.

Given our bias toward three-dimensional representations, we prefer volume-oriented formalisms over surface-oriented ones, as volumetric representations support clean decomposition of objects into natural parts (i.e., parts correspond to volume elements, defined by continuity, rather than to surface elements). They can also give access to quasi-invariants that reduce the computational complexity of matching and that simplify discrimination between an image's figure and ground.

One approach that has these desirable characteristics, and which has received considerable attention in the vision community, describes complex objects as combinations of *generalized cylinders*, a three-dimensional formalism that represents each component of the object in terms of a two-dimensional cross section that is swept through a path. We limit ourselves here to homogeneous generalized cylinders, a special class in which the cross section has constant shape and orientation, but can vary in size over the course of the path; thus, the sweep function specifies only scaling information.

Following Binford, Levitt, and Mann's (1989) previous work in this area, we describe each generalized cylinder in terms of three functions: a cross section, an axis function, and a sweep function. The cross section is a surface that typically specifies the shape of a cut made perpendicular to the major axis of the object, the axis function describes the three-dimensional path of that axis, and the sweep function specifies the transformation of the cross section along the length of the sweep. Figure 1 shows the wing of an airplane, along with the cross section, axis function, and sweep function for this structure.

There are many different ways to represent functions of this sort internally. For instance, we might describe each function using a piecewise linear or piecewise quadratic function, or we might use a single higher-order polynomial. Here we use instead a sequence of points. For the cross section, the first point corresponds to some arbitrary position along the curve; the axis function begins at one end of the object; and the scaling function indicates size as a function of distance along the axis.

The above formalism lets one represent a rich set of primitive objects, but more complicated structures require an extended language. We describe each complex object as a compound of generalized-cylinder parts, along with a set of relations among those parts. Although we could specify these spatial relations in terms of the components' positions and orientations within a common coordinate system, we instead use higher-order relations that are invariant across different perspectives.

In particular, we select one component P (the one with the longest axis, which is likely to be visible from most perspectives) as primary for a given complex object and compute its spatial relations with each non-primary component C. In particular, we compute the straight-line vector P_A between the first and last points along the primary P's axis, and we compute an analogous vector C_A for each nonprimary component. We then compute the dot product of these two vectors to determine the angle between the two components' axes. This scalar takes on values between zero (when C and P are parallel) and one (when C and P are perpendicular). Consider the plane depicted in Figure 2; here the angle between

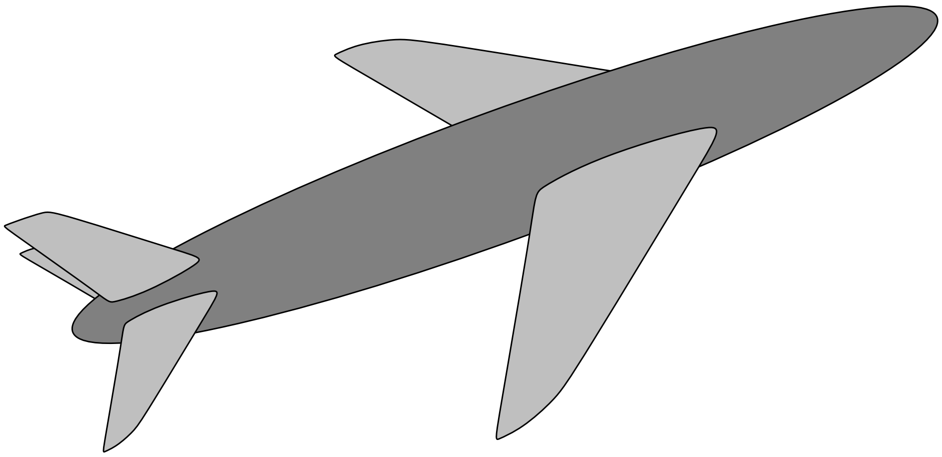

Figure 2: An airplane that can be described in terms of six component parts (the fuselage, two main wings, two tail wings, and a tail), each characterized in turn as generalized cylinders. The orientations and relative sizes of these components provide important constraints on the concept.

the axis of the plane's fuselage (the primary component) and the axis of its left wing is 60 degrees, giving 0.866 as the dot product. A more swept-back wing would have a lower value.

The above term describes the spatial relations between the component objects' axes, but not the relations among their cross sections. For the primary component, we use a similar measure found by computing the straight-line vector P_C that starts at each cross section's center (measured at the first point along the axis) and ends at the first point describing the cross section. As before, we compute the analogous vector C_C for each nonprimary component, then calculate the dot product for each pair, which varies from zero (when the cross sections are oriented in the same direction) and one (when they are perpendicular). For the plane in Figure 2, this term is needed to specify the relative orientation of the two wings, to ensure they point in the same direction.

Finally, we characterize the relative sizes of object components in terms of the lengths of their axes. Specifically, for each nonprimary component C we compute the ratio P_C/P_A, using the same vectors we computed to describe the angles between component axes. This scalar can range from $-\infty$ to ∞, though it will often be less than one because the primary component will be larger than other parts. For instance, the axis length for the plane fuselage in Figure 2 is longer than those for the wings or tail.

3. Representing Classes in Memory

In the previous section we described our representation for objects that appear in images. However, to support recognition and reconstruction, long-term memory must contain more than descriptions of individual objects; it must also store descriptions of object *classes*. Such descriptions must specify both the structure held in common among members of each class and their inherent variation. For example, the typical plane includes a fuselage, two major wings, two tail wings, and a tail in roughly the same configuration, but the details of these components and their spatial relations vary considerably.

Researchers have developed a number of methods for representing abstract classes. One common approach is to use a logical description that specifies a set of sufficient features or relations for membership in a class. However, we favor a probabilistic framework that lets one take uncertainty into account during the inference process. One common organization for probabilistic knowledge is known as a Bayesian network (Charniak, 1989). This framework assumes a set of nodes, representing attributes or variables, connected by a set of directed links, indicating causal relations among the attributes. Stored at each node is a table that specifies the conditional probability distribution for the values of that attribute for each combination of values of its parent attributes. An absence of links between two attributes indicates that they are conditionally independent given their parents.

Figure 3 shows the structure of the Bayesian network we use to encode three-dimensional models of object classes. The top node specifies the class of the composite object, such as plane or rocket launcher. This node has two general types of children. One type (the black circles) represents the classes of component objects, such as wings and turret or fuselage and base. The other (black squares) specifies spatial relations among these components, such as the dot products and ratios described in the previous section. These nodes differ from the composite and component nodes in that they store Gaussian distributions (in terms of means and variances) over numeric attributes, rather than discrete probability distributions over nominal variables.

The component nodes in turn each have three children (shown as white circles). One of these represents the possible cross sections, another the alternative scal-

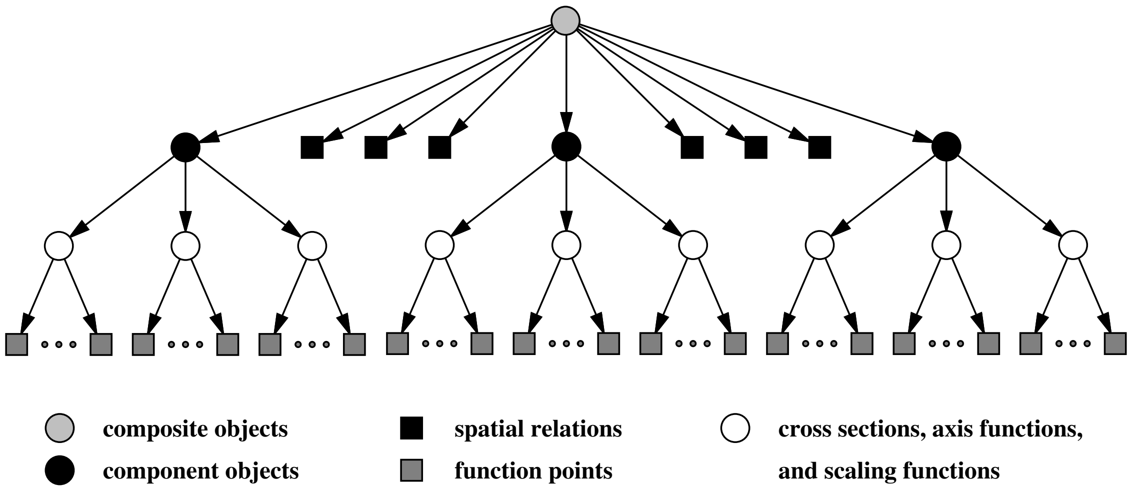

Figure 3: Structure of the Bayesian network used to encode knowledge of object classes at different levels of aggregation. The top level describes classes of composite objects, the second level summarizes their components (black circles) and the spatial relations among them (black squares), the third level (white circles) represents the cross sections, axis functions, and scaling functions of generalized cylinders, and the lowest gives the points used in those functions. Circles depict nodes for discrete variables, whereas squares stand for continuous variables.

ing functions, and another the possible axis functions that can occur for component objects. For now, we have chosen to treat these as nominal attributes, specifying a small set of discrete alternatives in each case. Thus, for axis functions we might have one class that corresponds to a straight line (for the fuselage) and another that denotes a straight line with a downward turn at the end (for wings). Similarly, for cross sections we might have a circle, an ellipse with major axis twice the length of its minor axis, and a rectangle with the same proportions.

The final level of the Bayesian network corresponds directly to the numeric variables used to describe each point in a cross section, axis function, or scaling function. Again, the conditional probability table stored with each node specifies a multivariate Gaussian distribution, described in terms of means, variances, and covariances; the latter are needed because there may exist correlations among variables not covered by the parent node.

We should note that the structure in Figure 3 constitutes a special form of Bayesian network, in that each node has exactly one parent. This means that the probability distribution for each attribute is directly dependent on only one other variable. As we will see in the sections that follow, this assumption considerably simplifies both learning and performance. In fact, each set of nodes and their common parent have the form of a naive Bayesian classifier (Langley, Iba, & Thompson, 1992), a simple probabilistic representation that assumes a set of predictor variables that are conditionally independent given the class attribute. Because our organization for visual knowledge combines a number of such entities[1] in this hierarchical manner, we will refer to this structure as a *cascaded Bayesian classifier*.

[1]The only minor exception is the lowest layer, in which we include covariances to handle possible correlations between pairs of variables.

4. Recognizing Classes of Objects

In previous sections we presented our formalism for describing objects in an image and our Bayesian network representation for object classes. Now we can examine the use of these descriptions in the processes of recognition and reconstruction. Briefly, given output from an early vision system in terms of generalized cylinders and spatial relations among them, we want to use the Bayesian network to classify the objects they describe and to infer the shape, position, and orientation of any occluded components.

Let us focus first on the classification process, reviewing the behavior of naive Bayes, as it plays a central role in the cascaded Bayesian classifier. The aim of the naive Bayesian algorithm is to determine the most likely class C given an observed test instance I. To this end, we apply Bayes' theorem to determine the probability of each class C_i given the instance, giving:

$$p(C_i|I) = \frac{p(C_i)p(I|C_i)}{p(I)} \ ,$$

where $p(C_i)$ is the prior probability of class C_i and $p(I|C_i)$ is the probability of the observed instance given this class. However, since I is a conjunction of j values, we can expand this latter expression to:

$$p(C_i|\bigwedge v_j) = \frac{p(C_i)p(\bigwedge v_j|C_i)}{\sum_k p(C_k)p(\bigwedge v_j|C_k)} \ ,$$

where the denominator sums over all classes and where $p(\bigwedge v_j|C_i)$ is the probability of the instance I given the class C_i. After calculating these probabilities for each class, the naive Bayesian classifier assigns the instance to the class with the highest overall probability.

In order to make the above expression operational, we must still specify how to compute the term $p(\bigwedge v_j|C_k)$. Because naive Bayes assumes independence of the components given the class, we can use the equality:

$$p(\bigwedge v_j|C_k) = \prod_j p(v_j|C_k) \ ,$$

where the values $p(v_j|C_k)$ represent the conditional probabilities stored with each attribute (node). This approach greatly simplifies the computation of class probabilities for a given observation.

Now that we have reviewed naive Bayes, we can examine the extensions necessary to support the cascaded version. We will first consider the top level of the Bayesian network and work downward from there. We want to compute $P(O_i|\bigwedge C_j)$, where O_i is the class of the composite object, and the 'instance' consists of the component classes C_1 through C_n, along with the spatial relations among them. For example, we might want to distinguish between a plane and a rocket launcher, or between different types of planes. Again, because we assume that the component classes and orientations are independent given the object class, we can simply take the product of the various conditional probabilities when determining the probability of the object class, using the expressions given above.

However, we must modify the naive Bayesian scheme somewhat because earlier stages of the object recognition process do not provide the class of each component, but rather a probability distribution across the classes. That is, we cannot tell for certain whether a component is a fuselage or a turret, but we do have probabilities for each such component class. In response, we use these distributions to compute the probability $p(\bigwedge C_j)$ of each combination $\bigwedge C_j$ of component classes, then compute:

$$\sum p(\bigwedge C_j) P(O_i|\bigwedge C_j) \ ,$$

which sums over all such combinations, to find the overall probabilities of the composite object classes O_i. Using such a sum, weighted by probabilities, is the standard Bayesian approach to dealing with uncertain situations.

We have shown the calculations needed to determine the probabilities of each object class (and thus to select the most likely one), but they rely on probabilities about the component classes and their relative spatial orientations. As described in Section 2, the orientation information can be computed directly from the descriptions provided by the early vision system, in that we can compute the dot products and ratios for each pair of components, then determine the probability $P(v_j|O_j)$ of the resulting values according to the conditional Gaussian distribution associated with each numeric attribute. Thus, we can determine the extent to which two components have the right orientations and sizes to serve as the fuselage and wing of a plane.

We must still compute the probability distributions for the component classes themselves, but we can apply the modified naive Bayesian approach recursively, as the structure of the hierarchy indicates that the cross section, axis function, and scaling function are independent given the component class. Thus, we have:

$$p(C_j|X, A, S) = \frac{p(C_j)p(X|C_j)p(A|C_j)p(S|C_j)}{\sum_k p(C_k)p(X|C_k)p(A|C_k)p(S|C_k)} \ ,$$

where C_j is the component class (e.g., fuselage or wing), X is the cross section (e.g., circle or ellipse), A is the axis function (e.g., straight or downturning), and S is the scaling function (e.g., constant or shrinking). As before, we must modify this somewhat to compute a sum weighted by the probability for each combination of values for the three generalized-cylinder functions.

The above calculations suppose a specific mapping between image components and model components. Given the same number of components in the model and image, for N components there are $N!$ possible mappings. Because N is typically small, we simply compute the posterior distribution for each of these mappings, select the mapping that produces the lowest mean-squared errors for the axis functions, cross sections, and scaling functions, and ignore the alternatives.[2] For example, given the image of a plane with six components, this scheme leads to 120 possible mappings, only one of which should have a low error. Another approach would compute a weighted combination of the probability distributions for each mapping, but this would require some way to determine the probability of each mapping, which we lack.

Finally, the recognition system must compute the probabilities for each type of cross section, scaling function, and axis function from the output of the early vision system, which produces an ordered set of points in 2D or 3D space for each function. We transform these observations into probabilities using a three-step procedure, which maps the observed points onto the points stored with each type:

- Calculate the distances between successive points in each image function and divide the cumulative distances by their sum to give fractions of arc length, scaling the values of the points by the same amount;
- Interpolate points in each image function to ensure the image function contains the same number of points, at the same fraction of arc length, as each corresponding model function;
- Compute the least-squares equation relating the image and model functions, weighted inversely by the variance of each model point;[3] this process rotates and translates the coordinate system of the image function to give the best fit to the model function;
- Use the transformed points to give values for the model variables, then use the naive Bayesian scheme (augmented by covariances) to compute the probability of each function type.

[2] More efficient methods that avoid these combinatorics are certainly possible, and we plan to incorporate one of them into future versions of the system.

[3] For the cross section, we find the least-squares equation using every possible image point as the mapping onto the first point in the model function, then select the one that gives the best fit.

This multistep process transforms the cross-section points for each image component into a probability distribution over the possible cross sections, and produces similar distributions for each axis and scaling function. Combined with the stages discussed earlier, it grounds the probability computations in a description of the image, and thus lets the system classify the object that appears in the image.

Clearly, the above scheme relies centrally on the extraction of generalized cylinder descriptions for objects and their components in the image. To this end, we plan to invoke software described by Zerroug and Nevatia (1994), which produces cross sections, axis functions, and scaling functions in the format we described earlier (i.e., as sets of points). Although we have not yet connected our recognition system directly to Zerroug and Nevatia's software, due partly to the technical difficulty of grouping edgels and generating generalized-cylinder descriptions, establishing this link has a high priority within our research program.

One can also use the cascaded structure for reconstruction rather than recognition. Suppose some image components are occluded, so that the early vision system produces descriptions for only some of them. One can still use this partial set to determine the probabilities for each composite object class, through the same mechanisms described above. One can then use this information to infer the most likely identities for the missing components, along with their orientations relative to the observed components. This inference process follows somewhat different lines from the classification process, but it can be carried out using standard algorithms for Bayesian networks. We have not yet implemented this reconstruction process, but we plan to incorporate it into future versions of the system.

Our approach to image understanding borrows heavily from Binford, Levitt, and Mann's (1989) work, which also combines a generalized cylinder representation for objects with a Bayesian network for object recognition and reconstruction. However, their framework differs from ours in two important respects. First, their Bayesian network deals with both early and late visual inference, extending from edgels in the input image, through curves and ribbons (in which quasi-invariants play a central role), to generalized cylinders and complex objects. In contrast, we have focused on the last few stages in order to simplify matters for learning. Second, Binford et al.'s approach constructs the Bayesian network dynamically, from the bottom up, on each step selecting the most likely candidates to extend further. This technique lets their system deal with quite complex images in which many apparent edges play no part in the final object description. By comparison, we have assumed a fixed network structure that assumes generalized cylinder descriptions are already available.

Liang, Christensen, and Jensen (1994) describe another approach that relies on Bayesian networks for 3D recognition and reconstruction, but that introduces aspect graphs or characteristic views as an intermediate stage between 2D curves and full 3D descriptions (for which they use geons rather than generalized cylinders). As in Binford et al., their method can use the inference network to perform bottom-up recognition from the image, top-down reconstruction from the models, or a mixture of these processes. The system also incorporates a decision-theoretic utility function, similar to that reported by Levitt, Binford, and Ettinger (1989), to direct the inference process and focus attention. Rimey and Brown (1994) also use this idea in their 2D system for detecting the location of objects. Clearly, our own work is most closely related to Binford et al.'s framework, though it is somewhat simpler due to our concern with learning issues.

5. Learning Models of Object Classes

As described by Langley et al. (1992) and others, learning in the naive Bayesian framework involves the simple process of incrementing a count each time the system encounters a new instance, along with a separate count for a class each time it observes an instance of that class. Together with the prior probabilities discussed below, these counts let the classifier estimate $p(C_k)$ for each class C_k. In addition, for each instance of a class that has a given nominal value, the algorithm updates a count for that class-value pair. Together with the second count, this lets the classifier estimate $p(v_j|C_k)$. For each numeric attribute, the method retains and revises two quantities, the sum and the sum of squares, which let it compute the mean and variance for a normal curve that it uses to find $p(v_j|C_k)$; a similar calculation lets it update the quantities needed to compute the covariance matrix (Suppes & Liang, 1995) if deemed necessary. Because some instances may have missing attributes, the system must include a fourth count for each class-attribute pair.

The hierarchical structure of the cascaded Bayesian classifier requires some extensions to this learning method. The most obvious is that it must update counts for conditional probabilities at every level of the structure. One response relies on the teacher to provide class labels not only for the composite object, but also for its components and their functions. This *supervised* approach effectively transforms the induction task of cascaded Bayes into a set of relatively independent naive Bayes tasks. This method requires more user attention than we would prefer, but it provides a good baseline.

We have also explored two semi-supervised techniques that only assume class labels for composite objects and numeric values that describe each generalized-cylinder function. One scheme takes a *competitive* approach that simply selects the most likely value for each node in the hierarchy, given the data, and updates its count by one. The other approach uses a *proportional* method that updates the count for each value by a fraction equal to the inferred probability for that value. The first technique is similar to methods for competitive learning in neural networks; the second relies on the more 'proper' Bayesian idea of operating directly on probability distributions. Analogous methods are possible for numeric attributes, but in the current system these are all 'observed', in that they are computed directly from the output of the early vision module.

Table 1: Components and functions for object classes used in experimental study.

Composite	Component	Cross Section	Axis Function	Scaling Function
Plane	Fuselage	Circular	Straight	Up-level-down
	Left-main-wing	Elliptical	Down-turning	Decreasing
	Right-main-wing	Elliptical	Down-turning	Decreasing
	Left-tail-wing	Elliptical	Down-turning	Decreasing
	Left-tail-wing	Elliptical	Down-turning	Decreasing
	Tail	Elliptical	Down-turning	Decreasing
Tank	Tank-Base	Elliptical	Straight	Constant
	Turret	Circular	Straight	Up-level-down
	Cannon	Circular	Straight	Constant
Building	Building-base	Rectangular	Straight	Constant
	Building-roof	Rectangular	Straight	Constant

Both the competitive and proportional methods lack one desirable feature of naive Bayes – its independence of training order (Langley, 1995). Because early revisions can influence the probabilities generated for later training cases, the order of presentation can affect the probability estimates stored with each node. To offset this tendency, one can run the training data through the learning algorithm repeatedly, as in some neural network methods, until no significant changes occur in the resulting probabilistic descriptions.

We have not yet explained how to transform the stored counts into probabilities, for use during classification. The most straightforward scheme for estimating $p(C)$, $p(v|C)$, and related terms simply computes these probabilities as ratios of the counts; for example, $p(C)$ would be the number of instances with class C divided by the total number of instances. However, this approach can lead to zero probabilities that can overwhelm other terms in the products computed during classification. Clark and Niblett (1989) describe one response to this problem; when no instances of a value have been observed, they replace the zero probability with $p(C)/N$, where N is the number of training cases.

Another way to avoid this problem is to specify prior knowledge about each probability distribution. One common scheme makes use of 'uninformed priors', which assign equal probabilities to each possible class and to the possible values of each attribute.[4] However, one must also specify how much weight to give these priors relative to the training data. For example, Anderson and Matessa (1992) use a Dirichlet distribution to initialize probabilities and give these priors the same influence as a single training instance, and we have used the same scheme in our implementation. Following their lead, we use an analogous technique to initialize the distributions for numeric variables.

[4]These priors concern the distribution of probabilities to be estimated during learning, and it is important not to confuse them with the priors used during the classification process, which themselves result from learning.

6. Experimental Studies of the Approach

We have posited two important characteristics of our approach to learning object models: that the induction process can improve the recognition of object classes based on experience and that this process can be aided by background knowledge stated in terms of generalized cylinders and Bayesian networks. However, these claims are actually hypotheses that, ultimately, can only be evaluated empirically. In this section we present two studies designed for this purpose.

Experiments in visual learning, as in other areas, involve some dependent variable that measures behavior along a dimension of interest and one or more independent variables that, when varied, might affect that behavior (Kibler & Langley, 1988). In this case, our dependent measure is the recognition or classification accuracy of the visual system; that is, the percentage of objects correctly assigned to their proper class. The independent variables include the number of training objects (to test the hypothesis that accuracy improves with experience) and the amount of background knowledge (to test the claim that this knowledge aids learning). Another independent variable of interest concerns the amount of noise in the domain, as reflected by the variation of objects within each class.

Because we had not yet interfaced with Zerroug and Nevatia's software, which will provide us with generalized cylinder descriptions of the objects in an image, we constructed a generator that produces synthetic data in the same format. Given a set of 3D 'target' models for different object classes, this generator can create random instances of each object class. Each instance is described as a set of components with associated cross sections, axis functions, and scaling functions (described in turn as sets of points), along with spatial relations among the components. Numeric values are sampled randomly from the Gaussian distribution specified in the target model, whereas nominal values are sampled randomly from discrete distributions. The generated object's ab-

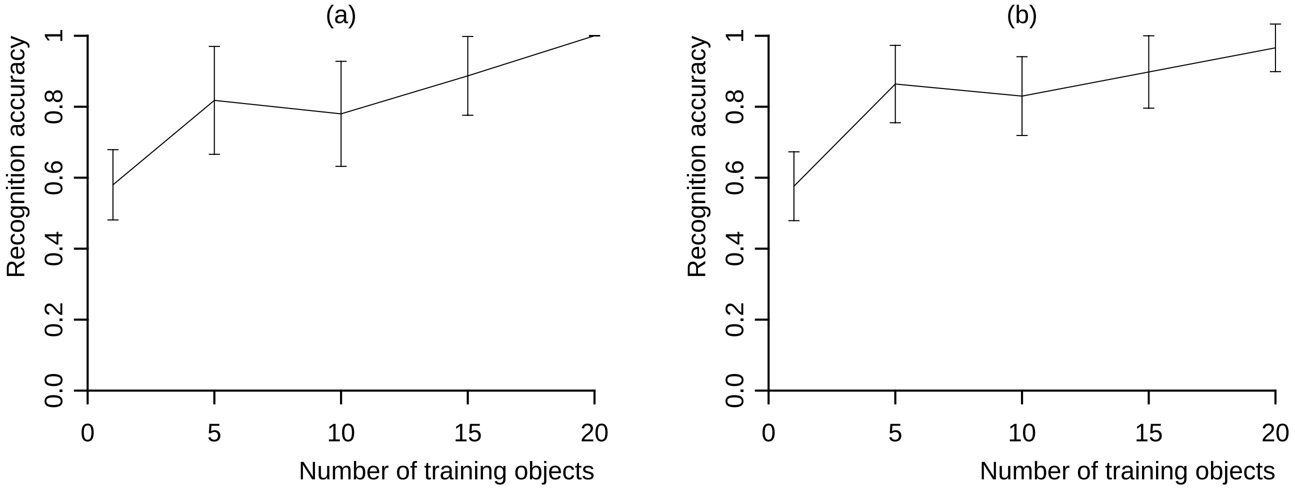

Figure 4: Learning curves on synthetic 3D image descriptions (a) with uninformed priors and (b) with background knowledge about the domain. The error bars represent 95% confidence intervals.

solute position and orientation in space are determined by randomly selecting a point in space from which the object is viewed, sampled from a region that lies greater than one distance from the object and less than another.

For the present studies, we handcrafted target models for three object classes: planes, tanks, and buildings, each with between two and six components. Table 1 presents the generalized cylinder components for these classes and the functions that describe them. The numeric variables (not shown in the table) are specified as the means and standard deviations for a conditional Gaussian. We used these target models to generate randomly ten sets of 20 training cases each and another 50 cases as a separate test set.

Our first experiment systematically varied the number of training objects available during learning. More specifically, we ran the supervised algorithm on the first five cases in each of the ten training sets, then over the first ten cases, and so forth. For each situation, we measured the accuracy of the learned Bayesian network on the 50 test cases, averaging the results across the different training sets. In each run, we initialized the cascaded Bayesian classifier with uninformed priors over all the classes that occurred in the target categories, to minimize knowledge of the domain.

Figure 4 (a) presents the resulting learning curve, which shows the mean accuracies and 95% confidence intervals for each experience level. The curve shows clearly that cascaded Bayes does improve its ability to recognize objects as it gains experience, at least in this synthetic domain. This result is not especially surprising, but the machine learning literature does contain cases in which performance actually degrades with increasing experience, so this basic experiment was necessary to counter that possibility.

Our second experiment was designed to show that background knowledge, in the form of a cascaded Bayesian classifier that encodes information about generalized cylinders, can aid the learning process. The motivation for this study comes from the notion that a developer may be able to enter models into an object library, but that only the gross model characteristics are likely to be accurate. The learning system should not only be able to use training data to revise the models, but should use this approximate knowledge to give more accurate recognition than a system that lacks it.

To test this idea, we repeated the conditions of the first study but, rather than using uninformed priors to initialize the Bayesian network, we used probability distributions based on the target models. In particular, we entered the accurate probability distributions for all discrete variables, but we left the numeric variables as in the previous study. Figure 4 (b) shows the learning curve that results when such background knowledge is present. Although accuracy is slightly better early in learning, the general behavior is surprisingly similar to that without background knowledge, presumably because the availabity of class information about all levels of the object is enough to constrain the learning process. We hypothesize that informed priors will provide more benefit to the semi-supervised methods, but testing that prediction must await future research.

7. Related Work on Visual Learning

Our approach to object recognition and visual learning has similarities to earlier work in this area but also some important differences. Here we briefly consider these other efforts, in each case discussing the representation and organization of knowledge, the performance element that uses that knowledge, and the method for acquiring it. We will see that most researchers have relied on clustering rather than supervised learning methods and that, although probabilistic descriptions have been common, they are typically organized by means other than

Bayesian networks.

For example, Sengupta and Boyer (1993) have (like us) taken an approach that represents objects models as probabilistic summaries at different levels of aggregation. However, they further organize these models in an is-a hierarchy, through which the recognition procedure sorts new descriptions. This sorting process leads to updates in the probabilistic summaries through which the description passes, and creates a new subclass upon finding children with class summaries that are sufficiently different from the new description. This incremental, unsupervised scheme is very similar to our earlier work on unsupervised concept formation (Gennari, Langley, & Fisher, 1989), differing primarily in its evaluation metric and its reliance on beam search for sorting rather than a greedy method. Sengupta and Boyer have tested their approach using descriptions taken from a CAD library of 3D objects.

Conklin (1993) describes another approach to learning visual categories that organizes memory in terms of an is-a hierarchy, but in which each nonterminal node contains not a probabilistic summary of training cases but logical conjunctions of features held in common by all of its children. These logical descriptions are designed to be transformation invariant, in that they remain true from different perspectives and distances. Conklin's system uses these invariant descriptions primarily as indices during retrieval of individual training cases, but also to help parse images as they are sorted through the hierarchy. As in Sengupta and Boyer's work, learning is incremental and interleaved with the sorting process, with training cases being stored as new terminal nodes but also leading to more general descriptions along the paths they traverse. Conklin has used his approach in the analysis of molecular scenes described as electron density maps.

Segen (1993) presents an alternative approach to visual learning which comes closer to our own, in that it uses probabilistic summaries for object models at different levels of part-of aggregation but only one is-a level. He describes each object class as a 'stochastic graph', which consists of a set of components, each specified with a discrete probability distribution over a set of nodes that are themselves stochastic graphs. The recognition process assigns an image to the most likely top-level graph, and the learning algorithm either incrementally updates the probabilistic summaries for the selected class or creates a new class if the image is different enough from existing ones. Segen's recursive stochastic graphs bear a strong similarity to our cascaded Bayesian classifier, though his unsupervised learning algorithm, which incorporates notions of minimum description length, differs from our semi-supervised techniques, which use a maximum-likelihood approach. Moreover, his representation for objects is two-dimensional rather than three-dimensional, as in our framework. Segen has tested his approach in the domain of gesture recognition.

Additional work on learning object models through clustering, reported by Gros (1993), takes a nonincremental, agglomerative approach. As in Segen's method, the basic representation is two-dimensional, in this case describing each class of objects as a set of characteristic views. Each view is summarized as a set of features, such as line segments and their points of intersection, that occur at a single level of aggregation. Gros does not describe a performance element, but one might use a variant on the nearest neighbor algorithm to to assign images to the characteristic view with which it shares the most features, and thus to an object class. The learning system relies on an unsupervised clustering algorithm that successively merges the two clusters of images that are nearest in the feature space, followed by postprocessing that uses a threshold to determine top-level classes.

Pope and Lowe's (1993) approach is similar to Segen's, in that they represent objects as sets of 2D characteristic views, each described as a set of features at multiple levels of aggregation. Associated with each feature is a probability of occurrence and a probability distribution for its numeric attributes. Recognition of new images involves using Bayes' rule to compute the probability of each view given the features found in the image, then selecting the most likely one. The learning scheme incrementally assigns each image description to the most likely view and updates the probability distributions for that view, but creates new characteristic views for sufficiently novel instances.[5] Although superficially different from our approach in that it operates on clusters rather than on a Bayesian network, the basic induction method is very similar to our own, except that it does not take advantage of class labels on training cases. However, the representational differences are more profound, with Pope and Lowe relying on 2D characteristic views rather than the 3D generalized cylinders that are central to our own work.

The importance of background knowledge to our approach distinguishes it from most work on vision and learning, but Cook, Hall, Stark, and Bowyer (1993) describe another method that builds on these ideas. They present their learning system with initial models for object classes and a set of inference rules for predicting the degree to which an object in an image satisfies some 'function'. Training cases have functionality scores, which the learning algorithm uses to revise the conditions on its inference rules, using a method similar to backpropagation in neural networks. Background knowledge lets the system infer 3D descriptions from images and also constrains the learning process. Their approach differs from ours in the use of inference rules to determine the degree to which a class is satisfied, as opposed to our Bayesian inference scheme, but the general style of encoding background knowledge, and its role in biasing the induction process, are very similar. Cook et al.'s approach also bears a close relation to earlier work by Winston, Binford, Katz, and Lowry (1983) on learning recognition rules from functional knowledge.

In summary, the recent literature reports a number of research efforts that address issues similar to those with which we are concerned. Many of these employ probabilistic representations to handle the uncertainty

[5] In related work, Beis and Lowe (1993) have focused on creating is-a hierarchies and indexing object models, which comes closer to the approaches taken by Conklin and by Sengupta and Boyer.

inherent in vision, and some describe complex objects at multiple levels of aggregation. However, none of these projects have used generalized cylinders to represent three-dimensional object models, nor have they taken advantage of Bayesian networks to structure this knowledge and constrain the learning process.

8. Conclusions

In this paper we described an approach to visual learning that draws on earlier work in both image understanding and machine induction. Our system characterizes objects in images in terms of generalized cylinder parts and relations among them, while it stores long-term knowledge about object classes in a Bayesian network. The recognition process relies on early vision software, developed by Zerroug and Nevatia (1994), to infer generalized cylinder descriptions from images, followed by probabilistic inference over the Bayesian network to determine the most likely cross sections, axis functions, scaling functions, component classes, and composite classes. The structure of the Bayesian network, which assumes independence of variables given their parents at each level, simplifies the learning process, which consists of updating counts for the most likely candidate at each level.

We reported preliminary results with this approach to learning object models, using synthetic but realistic data to show that the system improves its recognition accuracy with experience and that it can take advantage of partial background knowledge. However, more work clearly remains to be done. In future research, we plan to test the system on output from Zerroug and Nevatia's software to measure robustness on actual images. We will also carry out experiments with additional synthetic data that involve greater within-class variation, thus testing our probabilistic inference and learning mechanisms under more challenging conditions. We also intend to compare experimentally the behavior of the competitive, proportional, and supervised methods for learning. Finally, we should evaluate our approach on recognition tasks involving hundreds of object classes, to test our claim that automated acquisition of model libraries is possible in such situations. Although our studies will undoubtedly reveal problems with the system, they should also suggest improvements, and we have high hopes that the basic framework will prove useful for a variety of image-understanding domains.

Acknowledgements

We thank Tod Levitt, Wallace Mann, Kurt Huang, and Walter Tackett for useful discussions on the ideas reported in this paper. We also thank Kurt Huang for writing the data generation code used in the experiments and Wallace Mann for making available software used in the matching routines.

References

Anderson, J. R., & Matessa, M. (1992). Explorations of an incremental, Bayesian algorithm for categorization. *Machine Learning*, *9*, 275–308.

Beis, J. S., & Lowe, D. G. (1993). Learning indexing functions for 3D model-based object recognition. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 50–54). Raleigh, NC: AAAI Press.

Binford, T. O., Levitt, T. S., & Mann, W. B. (1989). Bayesian inference in model-based machine vision. In L. N. Kanal, T. S. Levitt, & J. F. Lemmer (Eds.), *Uncertainty in artificial intelligence* (Vol. 3). North Holland.

Charniak, E. (1989). Bayesian networks without tears. *AI Magazine*, Winter, 50–63.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, *3*, 261–284.

Conklin, D. (1993). Transformation-invariant indexing and machine discovery for computer vision. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 10–14). Raleigh: AAAI Press.

Cook, D., Hall, L., Stark, L., & Bowyer, K. (1993). Learning combination of evidence functions in object recognition. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 139–143). Raleigh, NC: AAAI Press.

Dickinson, S., Pentland, A., & Rosenfeld, A. (1992). 3-D shape recovery using distributed aspect matching. *Pattern Analysis and Machine Intelligence*, *14*, 174–198.

Gros, P. (1993). Matching and clustering: Two steps towards automatic object model generation in computer vision. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 40–44). Raleigh: AAAI Press.

Gennari, J. H., Langley, P., & Fisher, D. H. (1989). Models of incremental concept formation. *Artificial Intelligence*, *40*, 11–61.

Kibler, D., & Langley, P. (1988). Machine learning as an experimental science. *Proceedings of the Third European Working Session on Learning* (pp. 81–92). Glasgow: Pittman. Reprinted in J. W. Shavlik & T. G. Dietterich (Eds.) (1990), *Readings in machine learning*. San Francisco, CA: Morgan Kaufmann.

Langley, P. (1995). Order effects in incremental learning. In P. Reimann & H. Spada (Eds.). *Learning in humans and machines: Towards and interdisciplinary learning science*. Oxford: Elsevier.

Langley, P., Iba, W., & Thompson, K. (1992). An analysis of Bayesian classifiers. *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 223–228). San Jose, CA: AAAI Press.

Levitt, T. S., Binford, T. O., & Ettinger, G. J. (1990). Utility-based control for computer vision. In R. D. Schacter, T. S. Levitt, L. N. Kanal, & J. F. Lemmer (Eds.), *Uncertainty in artificial intelligence* (Vol. 4). North Holland.

Liang, J., Christensen, H., & Jensen, F. (1994). Qualitative recognition using Bayesian reasoning. In E. Gelsema & L. Kanal (Eds.), *Pattern recognition in practice* (Vol. 4).

Murase, H., & Nayar, S. K. (1993). Learning and recognition of 3D objects from brightness images. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 25–29). Raleigh, NC: AAAI Press.

Pope, A. R., & Lowe, D. G. (1993). Learning 3D object recognition models from 2D images. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 35–39). Raleigh: AAAI Press.

Rimey, R., & Brown, C. (1994). Control of selection perception using Bayes nets and decision theory. *International Journal of Computer Vision*, *12*.

Segen, J. (1993). Learning shape models for a vision-based human-computer interface. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 120–124). Raleigh, NC: AAAI Press.

Sengupta, K., & Boyer, K. L. (1993). Incremental model base updating: Learning new model sites. *Working Notes of the AAAI Fall Symposium on Machine Learning in Computer Vision* (pp. 1–5). Raleigh, NC: AAAI Press.

Suppes, P., & Liang, L. (1995). Concept learning rates and transfer performance of several multivariate neural network models. In C. Dowling, F. Roberts, & P. Theuns (Eds.), *Progress in mathematical psychology*. Mahwah, NJ: Lawrence Erlbaum.

Winston, P. H., Binford, T. O., Katz, B., & Lowry, M. (1983). Learning physical descriptions from functional descriptions. *Proceedings of the Third National Conference on Artificial Intelligence* (pp. 433–439). Washington, DC: AAAI Press.

Zerroug, M., & Nevatia, R. (1994). Three-dimensional part-based descriptions from a real intensity image. *Proceedings of the Image Understanding Workshop* (pp. 1367–1374). Monterrey, CA: Morgan Kaufmann.