Learning Indexing Functions for 3-D Model-Based Object Recognition

Jeffrey S. Beis and **David G. Lowe**
Dept. of Computer Science
University of British Columbia
Vancouver, B.C., Canada
email: beis@cs.ubc.ca
lowe@cs.ubc.ca

Abstract

Geometric indexing is an efficient method of recovering match hypotheses in model-based object recognition. Unlike other methods, which search for viewpoint-invariant shape descriptors to use as indices, we use a learning method to model the smooth variation in appearance of local feature sets (LFS). Indexing from LFS effectively deals with the problems of occlusion and missing features. The functions learned are probability distributions describing the possible interpretations of each index value. During recognition, this information can be used to select the least ambiguous features for matching. A verification stage follows so that the final reliability and accuracy of the match is greater than that from indexing alone. This approach has the potential to work with a wide range of image features and model types. A full 3-D recognition system has been implemented, and we present an example to demonstrate how the method works with real, cluttered images.

1 Introduction

Model-based object recognition consists of matching features between an image and a pre-stored object model. The hypothesize-and-test paradigm uses minimal sets of features to form correspondence hypotheses. From these, model pose is calculated, and model presence is verified by back-projection into the image and a search for further matches. The entire process is very expensive because each verification is expensive, and because of the computational complexity of generating hypotheses through exhaustive comparison.

Indexing is one way to combat this computational burden. It is a 2-stage process. At compile-time, shape descriptors derived from model feature sets are used to generate vectors corresponding to points in an index space. These are stored in some appropriate data structure for future reference. At run-time, the same process is used to generate index vectors from test images. The data structure is then used to quickly access nearby pre-stored points. Thus, correspondence hypotheses are recovered without comparing all pairs of model/image feature sets. As well, fewer and better hypotheses can be recovered by making the indexing more specific, by basing on more complex shape descriptors.

The information that we want to access with our indexing system is not just a number of correspondence hypotheses, but also the probability of each one being a correct interpretation of the data. To begin with, we form groupings of features exhibiting certain non-accidental properties. This is an essential step which reduces the number of indices that will be considered. These Local Feature Sets are then used to compute index values.

The first part of the actual indexing stage quickly recovers a set of nearest neighbors to each index value. We choose a tree structure (e.g. kd-tree) instead of the usual hash table to accomplish this (see section 3.1). In the second part of the indexing stage, indexing functions are applied to the neighbors in order to rank the hypotheses before sending them to a verification stage. The indexing functions are designed to interpolate between nearby views of the same model grouping, modelling the probability distribution over the possible interpretations of each index value. These rankings will in general mean that correct interpretations can be investigated first.

2 Previous Work

In this paper we are interested in indexing 3-D models from single 2-D images. Much work in the past on indexing has dealt with the easier problems of 2-D models [Knoll and Jain, 1986], [Wallace, 1987], [Stein and Medioni, 1992], or 3-D models where more information is available, such as with 3-D range data [Stein and Medioni, 1992] or multiple images [Mohan *et al.*, 1993].

When compiling an index, the most efficient storage possible can be achieved by using shape descriptors invariant with respect to viewpoint. For each underlying model feature set, only a single index value must be stored. It has been shown [Clemens and Jacobs, 1991],[Burns *et al.*, 1993], however, that no general, non-trivial invariant functions exist for 3-D point sets under the various standard projection models. This explains the two major strains of indexing approach that are present in the literature.

The first methodology involves adding constraints to create a restricted domain of 3-D feature sets that can be used to generate invariants. Forsythe, *et al.*, [1990] create invariant functions from grouped planar curves, such as coplanar pairs of conics or sets

of at least 4 coplanar lines. Lambdan and Wolfson [1988] also restrict to planar faces. They generate affine bases in which the coordinates of points on the face are invariant. [Rothwell *et al.*, 1992] generalize this work to deal with the more general perspective transformation.

Such invariants are very useful when available, but many objects will not contain one of these. A second set of approaches ignores invariants, and attempts to store an index value for each possible appearance of a feature set. These methods generate indices for multiple views of a model over some tesselation of the viewing sphere.

Thompson and Mundy [1987] use simple groupings of pairs of vertices to retrieve hypothetical viewing transforms for their models. Because of the lack of specificity in the indices, a voting scheme must be used to collect the noisy pieces of information into meaningful hypotheses. Clemens and Jacobs [1991] show that the view variation of indices derived from point sets generate 2-D sheets embedded in 4- (or higher-) dimensional index spaces. Jacobs [1992] provides a more space-efficient construction which reduces the 2-D sheets to two 1-D lines, each embedded in a 2-D space.

3 General Framework

3.1 Indexing

A common factor in all of these methods is that they discretize the indices and fill hash tables with the quantized values. The advantage is that the run-time indexing step can be done in constant time. The disadvantage is that the space requirements become excessive with even moderately complex feature groupings and moderate noise in the data. This arises due to the standard way these methods have dealt with noisy images. During table construction, an entry is made in each hash bin that is within an error radius ϵ of the actual model index values that are to be stored. Then all relevant hypotheses can still be accessed by looking in just one bin.

The number of entries is exponential in the index dimension, and the base of the exponential increases with noise and the required degree of specificity (more specificity equals finer bin divisions.) The only other way to deal with the noise issue is, for each image index value, to search the ϵ-volume of index space around that index at run-time, and then to form the union of the recovered hypotheses. This certainly mitigates the original advantage of the approach.

A second issue which is not adequately addressed by other methods, is the saliency of the indices. They generally treat all recovered hypotheses as being equally likely to be correct interpretations. Because the verification stage is relatively expensive, a heavy burden is placed on the indexing stage to provide only a small number of hypotheses. This in turn leads to the necessity of higher-dimensional and/or more finely partitioned index spaces, exacerbating the storage problem. We can do better by weighting each hypothesis both according to some measure of the proximity of the image indexing vector with stored model vectors, and according to some measure of the uniqueness of the possible interpretation.

Our approach addresses these two basic issues. During a learning stage, index vectors are generated from multiple views of the models. These are stored in a data structure which allows for efficient run-time determination of nearest-neighbors (NN). No extra storage is used to account for noise: a single datum is stored for each view of a model group. At the same time, a machine learning algorithm is used to tailor indexing functions, which will give us probability distributions over the set of potential interpretations stored in the index structure.

At run-time, a set of neighbors is recovered for each index vector computed from the test image. These are fed to the indexing functions, and the resulting probability estimates used to rank the hypotheses for the verification stage. Note that in creating the probability estimate for each hypothesis, our method effectively combines information from several sources.

Furthermore, the ranking step allows us to limit, in a principled manner, the time spent searching for objects in an image. Shapes which match too many of the pre-stored indices will only have a small likelihood of matching any single one of them. These (and others) might be dropped from further consideration by thresholding on the computed probability estimates. Or, we could choose to look at only a certain number of the most likely candidates before stopping. Either way, ranking serves as an extra filter on hypothesis generation, leading the whole recognition process to be more efficient.

3.2 Grouping

For indexing to be effective it is important that some data-driven grouping mechanism produce sets of features likely to come from a single object [Lowe, 1985]. Grouping can be based on certain non-accidental properties of images such as edge parallelism, co-termination, and symmetry. Properties are non-accidental if there is a high probability that they stem from the same underlying cause, i.e., from the same object.

These types of grouping could be said to have "qualitative invariance" in that, for example, segments co-terminating on a model will always project to co-terminations in an image. As such, they could be used as a crude form of indexing, but generate too many ambiguous hypotheses. For example, any set of three parallel edges found in an image might be used to match any three parallel edges from any model.

However, such groupings can be used to generate real-valued, multi-dimensional indices which can then be used to discriminate between groupings of the same type. Each index dimension will correspond to either some property of a particular feature in the grouping (e.g., a statistical value for a

texture region) or to a relationship between two or more of the features (e.g., angle between two edge segments). Each type of grouping will also have a canonical ordering of the derived features so that each dimension of the index has a consistent meaning.

As a further requirement, we stipulate that our groupings must be "local" within the images. "Local" will be some function of the size and separation of features in the set. Using Local Feature Sets for indexing increases robustness to occlusion and to missing features from either noise or the weaknesses of feature detectors.

3.3 Learning

As mentioned above, while the qualitative properties which are used to form feature groupings in our system are viewpoint invariant, the feature vectors that are derived from these groupings are not. Since we do not have invariant descriptors, each underlying model feature grouping generates not a single index value but a range of values occupying some volume of the index space. Changing one's viewpoint about a model grouping while all features of the grouping remain visible corresponds to moving about within that region.

As we do not have analytic expressions to describe these volumes, we will use a learning algorithm to approximate them. Formally, if $\vec{\mathbf{x}}$ is a local feature vector derived from an image and m is a correct match of features to a particular model, we want to learn $P(m|\vec{\mathbf{x}})$ for each possible interpretation m using a set of examples $(\vec{\mathbf{x}}, P(m|\vec{\mathbf{x}}))$.

Because the index values for a model grouping are a complex, non-linear function of projection from 3-D to 2-D, we require a method that can learn non-linear mappings. Poggio et. al. [1990] have used Radial Basis Function (RBF) networks to learn the continuous range of appearances of simple wire-frame models. While successfully demonstrating the interpolation ability of the networks, this work assumed the segmentation and occlusion problems had already been solved. Furthermore, the RBF was forced to make a yes/no decision on object presence/absence based only on a single feature vector derived from an image. That worked with simple models, but the probability of observing similar feature vectors from two different models increases rapidly with model complexity.

There is a more robust way to apply RBFs, as indexing mechanisms. With $\vec{\mathbf{x}}$ and m from above:

$$\vec{\mathbf{f}}_m(\vec{\mathbf{x}}) = \sum_i C_{mi} G(\vec{\mathbf{x}} - \vec{\mathbf{x}}_\mathbf{i})$$

where the G are the basis functions centered at the $\vec{\mathbf{x}}_\mathbf{i}$, $\vec{\mathbf{x}}$ are the test input vectors, and the C_{mi} are coefficients determined by the learning algorithm. In this paper, a simple form of RBF network is used: G are taken to be Gaussians; the $\vec{\mathbf{x}}_\mathbf{i}$ to be the set of training vectors; and the C_{mi} are computed by pseudo-inverse of the matrix with entries $G_{ij} = G(\vec{\mathbf{x}}_\mathbf{i} - \vec{\mathbf{x}}_\mathbf{j})$, where $\vec{\mathbf{x}}_\mathbf{i}$ and $\vec{\mathbf{x}}_\mathbf{j}$ are from the set of training examples (see [Poggio and Girosi, 1989] for details). Note also the hidden parameters $\vec{\sigma}$ of dimension equal to the index space, which we set by hand.

The RBF framework has the following advantages. Firstly, the nature of the learning algorithm is intuitively understandable. Our network is a linear combination of Gaussians centered on the training vectors. As we move away from an isolated training example in index space, the normalized activation determined by evaluating the RBF will fall smoothly from one to zero. The region between two closely spaced training vectors will contain interpolated activities.

Secondly, at a cost of increased complexity and (off-line) processing time taken by the learning algorithm, the system can be optimized by learning the σ for each dimension, and the positions of the basis vectors [Poggio and Girosi, 1990]. Knowing how important each dimension of the index space is for discrimination is important. It is likely even more important to know the variation of this factor *within* each dimension. We re-iterate that our approach does not rely on a single learning algorithm. Weighting the importance of the things we want to learn and fitting those to a choice of learning algorithm are topics for further work.

For the learning stage of our simple RBF implementation, synthetic images of 3-D object models are taken from various viewpoints. The training vectors are derived from the LFS groupings found in these images. For recognition, we use real images, and the RBF is evaluated for index vectors $\vec{\mathbf{x}}$ which are the nearest neighbors recovered by indexing with the image groupings.

The RBF output is an activation that is related to the probability that the image grouping corresponds to a particular model grouping. These hypotheses are then ranked according to their probability weightings. If the system is trained with a set of images representative of the test image domain, using both positive and negative examples, then the most salient groupings will indeed have the highest rankings. A rigorous verification stage (iterations of back-projection and match extension) ensures that the final reliability of interpretation is much higher than that of the initial RBF indexing.

4 Experiment

An example of the successful recognition process is presented (see Figure 1). For the learning stage, a hand-input model of a note holder was used. 72 synthetic images over a tesselation of a hemisphere were used to generate the indexing functions. Primitive features were straight-line segments, and the Local Feature Set groupings were chains of co-terminating segments (Figure 1(a)).

Although larger feature groupings do lead to more specific indices, there is a practical uncertainty principal between the specificity and the robustness of a grouping, due to noise and occlusion. In this ex-

ample, 4-segment chains were used, giving rise to 4-dimensional index vectors - three angles and the ratio of the interior edge lengths. These features are largely invariant to translations, scale, and image-plane rotations, which means that the learning algorithm only needs to model the variation in the few remaining degrees-of-freedom of the pose. A kd-tree was used to store the 10261 training vectors.

For recognition, the edge primitives were extracted from a real, cluttered image, taken by a CCD camera under no special lighting conditions. The processed image contains 297 segments which formed 155 groupings of 4-segment chains. Note that grouping of all combinations of segments in this cluttered scene would have produced over 10^8 unique collections of 4 segments, a number that would debilitate even the most accurate indexing mechanism. For each of the 155 index vectors, 50 nearest neighbors were recovered from the kd-tree and the RBFs evaluated for each of these.

This process generated generated 85 hypotheses. The top-ranked hypothesis (Figure 1(b)) is a correct one. It leads to the match in (Figure 1(c)). The top 4 ranked hypotheses were correct, as were 8 of the top 20.

References

[Burns *et al.*, 1993] J.B. Burns, R.S. Weiss, and E.M. Riseman. View variation of point-set and line-segment features. *IEEE Trans. PAMI*, 15(1):51–68, 1993.

[Clemens and Jacobs, 1991] D.T. Clemens and D.W. Jacobs. Space and time bounds on indexing 3-d models from 2-d images. *IEEE Trans. PAMI*, 13(10):1007–1017, 1991.

[Forsyth *et al.*, 1990] D. Forsyth, J.L. Mundy, A. Zisserman, and C.M. Brown. Invariance - a new framework for vision. In *Proceedings ICCV '90*, pages 598–605, 1990.

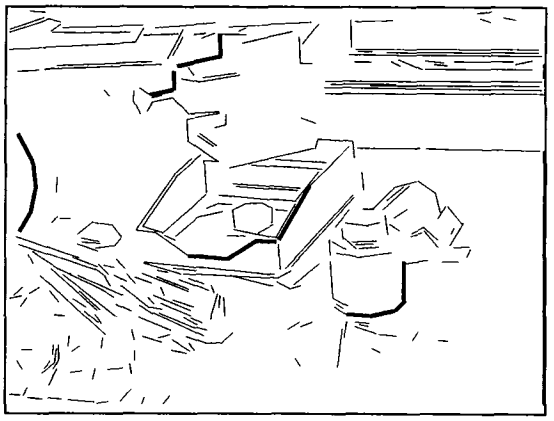
[Jacobs, 1992] D.W. Jacobs. Space efficient 3D model indexing. In *Proceedings CVPR '92*, pages 439–444, 1992.

[Knoll and Jain, 1986] T.F. Knoll and R.C. Jain. Recognizing partially visible objects using feature indexed hypothesis. *IEEE J. Rob. Aut.*, RA-2(1):3–13, 1986.

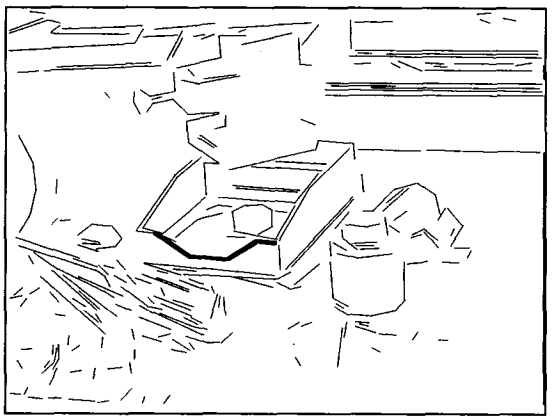
[Lambdan and Wolfson, 1988] Y. Lambdan and H.J. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *Proceedings ICCV '88*, pages 238–249, 1988.

[Lowe, 1985] D.G. Lowe. *Perceptual organization and visual recognition.* Kluwer Academic, Hingham, MA, 1985.

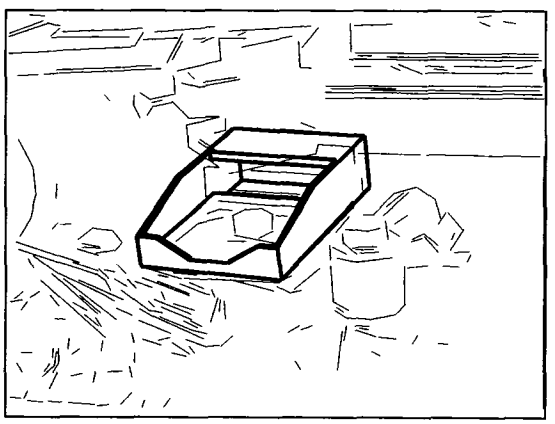
[Mohan *et al.*, 1993] R. Mohan, D. Weinshall, and R.R. Sarukkai. 3D object recognition by indexing structural invariants from multiple views. In *Proceedings ICCV '93*, pages 264–268, 1993.



(a)



(b)



(c)

Figure 1: (a) Examples of segment chains found by the grouping process. (b) The top-ranked hypothesis generated by the indexing process. (c) The final match of the model to the image, after the verification stage.

[Poggio and Edelman, 1990] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.

[Poggio and Girosi, 1989] T. Poggio and F. Girosi. MIT AI Memo, No. 1140, July 1989.

[Poggio and Girosi, 1990] T. Poggio and F. Girosi. MIT AI Memo, No. 1167, April 1990.

[Rothwell *et al.*, 1992] C.A. Rothwell, A. Zisserman, J.L. Mundy, and D.A. Forsyth. Efficient model library access by projectively invariant indexing functions. In *Proceedings CVPR '92*, pages 109–114, 1992.

[Stein and Medioni, 1992] F. Stein and G. Medioni. Structural indexing: efficient 2-D object recognition. *IEEE Trans. PAMI*, 14(12):1198–1204, 1992.

[Stein and Medioni, 1992] F. Stein and G. Medioni. Structural indexing: efficient 3-D object recognition. *IEEE Trans. PAMI*, 14(2):125–145, 1992.

[Thompson and Mundy, 1987] D.W. Thompson and J.L. Mundy. Three-dimensional model matching from an unconstrained viewpoint. In *Proceedings Int. Conf. Rob. Aut. '87*, pages 208–220, 1987.

[Wallace, 1987] A.M. Wallace. Matching segmented scenes to models using pairwise relationships between features. *Imag. vis. comput.*, 5(2):114–120, 1986.