

Index Trees for Efficient Deformable Shape-based Retrieval

Lifeng Liu and Stan Sclaroff

Image and Video Computing Group — Computer Science Dept.
Boston University — Boston, MA 02215 USA

Abstract

An improved method for deformable shape-based image indexing and retrieval is described. A pre-computed index tree is used to improve the speed of our previously reported on-line model fitting method; simple shape features are used as keys in a pre-generated index tree of model instances. A coarse to fine indexing scheme is used at different levels of the tree to further improve speed. Experimental results show that the speedup is significant, while accuracy of shape-based indexing is maintained. A method for shape population-based retrieval is also described. The method allows query formulation based on the population distributions of shapes in each image. Results of population-based queries for a database of blood cell micrographs are shown.

1. Introduction

Retrieval by shape is a key topic in content-based image retrieval research. Unfortunately, retrieval by shape requires object detection and segmentation. In previous work[11], we described a system that can detect, segment, and index multiple deformable shapes fully-automatically. The method can detect multiple shapes even in the presence of shadows or highlights, or when shapes touch. A limitation of the system was the amount of computation required to segment each image. The shape model fitting procedure, which tests the model against candidate region groupings, must be invoked to evaluate each possible image partition. Since fitting involves optimization, segmentation was slow for images of moderate complexity.

In this paper, we propose a method that uses pre-computation to accelerate the speed of on-line model fitting and image indexing. During on-line model fitting, simple shape features are used as keys in a pre-generated index tree of model instances. A coarse to fine index scheme is used to further improve speed while maintaining accuracy. As will be seen in the experiments, the proposed index tree structure provides nearly an order of magnitude speedup over the previous algorithm.

A method for shape population-based retrieval is also described. Histogram similarity measurements are used to compare the similarity of shape populations for different images. Results of population-based image queries for a database of blood cell micrographs are shown.

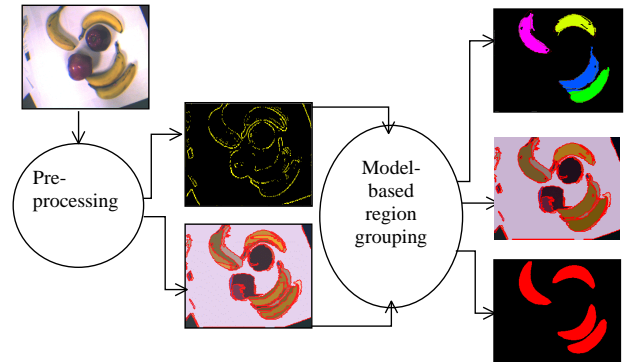


Figure 1: System overview. The original color image (image of bananas) undergoes pre-processing, which results in an oversegmentation and an edge map. These are inputs to the model-based region grouping stage (using a banana template). The final output includes region groupings for detected objects (four bananas), and recovered models for the objects.

2. Background

In [11] we proposed a method for deformable shape detection and description. We would like to review it briefly here. As show in Fig. 1, the deformable model-based segmentation system includes a pre-processing (over-segmentation and edge detection) stage, and a model-based region grouping stage. In the region grouping stage, for each potential region group, we define a cost:

$$\mathbf{E} = \mathbf{E}_{\text{color}} + \alpha \mathbf{E}_{\text{area}} + \beta \mathbf{E}_{\text{deform}} \quad (1)$$

where α and β are scalars that control the relative importance of the three terms: $\mathbf{E}_{\text{color}}$ is a region color compatibility term for the region grouping, \mathbf{E}_{area} is a region/model area overlap term, and $\mathbf{E}_{\text{deform}}$ is a deformation energy for the shape model. In our formulation, shape model is specified in terms of generic warping functions applied to template shapes, and a model fitting procedure is used to compute the cost in Eq. 1.

Further, in order to test the quality of a possible partitioning, a global cost function for partitioning the whole image is defined:

$$\varepsilon = \sum_{i=1}^n \mathbf{r}_i \mathbf{E}(\mathbf{g}_i) + \gamma \mathbf{n} \quad (2)$$

where γ is a constant factor, \mathbf{n} is the number of the groups in the current configuration, and \mathbf{r}_i is the ratio of i_{th} group

area to the total area. The cost value $E(g_i)$ for the group g_i is as defined in Eq. 1. We then use the highest confidence first algorithm to find the approximately optimal image partitioning. For a more detailed formulation and experiments, please see [11].

2.1. Performance of the Preliminary System

Experiments in [11] show that our preliminary system can detect touching deformable objects correctly and recover their shape descriptions. However, one problem with the preliminary system is that segmentation can be slow for images of moderate complexity. This is because the shape model fitting procedure must be invoked many times in order to get the cost values of different configurations. In the evaluation of each image partitioning (Eq. 2), we must compute the cost of every region grouping (Eq. 1) in the partitioning, which will invoke the model fitting procedure.

Deforming the model template to get a good fit is also an optimization problem. In the preliminary system, we used down-hill simplex method in fitting. The advantage of downhill simplex method is that it does not require explicit derivative computation, and can obtain good fitting in most cases; however, it is inefficient. For an image of 300x300 pixels, on an SGI O2, it may take over ten minutes to get the final result although we used an approximation method in the global optimization.

Our experiments' statistics show that, in the second stage of the system, model-based region grouping, CPU time for executing the shape model fitting procedure accounts for over 90% of the total CPU time. Although we have utilized some methods to speed up the fitting procedure, such as multi-resolution fitting, and caching deformation parameters, most of the CPU time is still used for the model fitting. We propose to use an index tree method to accelerate the model fitting procedure. As will be seen, this will speed up our system significantly.

3. Related Work

Many researchers have studied the object matching and recognition problem [8, 9, 10, 14]. One approach is to regard it as an optimization problem, and search in the model parameter space to find a match[14]. However, the computation complexity is prohibitive. Image-based object matching[13] is used to avoid searching the best parameters in the large space for model fitting. Another approach is geometric hashing[9].

We would like to handle the model fitting problem in a similar manner. As a pre-computation, we generate enough instances of the object class. Then, during on-line computation compare the region grouping with the instances to get the most similar one. However, the complexity of the algorithm may be linear to the number of object instances in the database, and computing the fitting costs for all the members in the database will be a big burden to the system. In [8] a method was proposed that uses a smaller number of models, but this approach assumes affine projection and

rigid objects.

We propose to organize the members in the database according to shape features so as to reduce the computation requirement. Tree structures are widely used in representing knowledge and decision rules, such as in interpretation trees[3]. We also use a tree structure for organizing the generated instances of the object class. However, unlike the interpretation tree where a different feature is used for correspondence in each level of the tree, in our index tree the whole shape feature vector is used in all the levels of the tree. One problem with a tree-structured code book search is that it does not in general find the nearest neighbor code vector [2]. We use a method that combines linear discriminant functions with neighbor subsets of deciding planes to deal with this problem.

From another view, our approach is also related to vector quantization(VQ) [2]. To minimize redundancy in the index, we store only the representative instances from the deformation space.

4. Basic Idea

The basic idea behind index trees can be explained as follows. We first generate a lot of deformed instances of the object class. We can generate a lot of deformed instances of the object class by sampling in the deformation space according to the prior distribution of the deformation parameters, as was obtained in the shape model training stage [11]. We then compute a shape feature vector for each generated instance. The shape feature vector and the deformation parameters are stored with the instance. This work can be done off-line as a pre-computation.

Then, in the fitting process, we compute the shape feature vector for a potential region group. Via comparing the shape feature vectors, the most similar one for the region group is fetched from the set of generated instances (called an **instance set**). Its associated deformation parameters are used as the parameters for the region group, or as a starting point to invoke a refining process. In order to speed up the search, we organize the instances in a tree structure, and call it an **index tree**.

5. Index Tree-Based Model Fitting

The requirement for model fitting is that it should be efficient and accurate. On the one hand, the number of pre-generated instances should be large and diverse enough to make the new fitting procedure robust. On the other hand, it should be fast to fetch the most similar instance. This brings the following problems:

Problem 1: How to form a set of instances such that it includes enough deformations and has little redundancy? For this problem, some attention can be paid to decrease the redundancy. First, instances can be obtained by randomly sampling in a bounded deformation space. The bounds are derived from the prior information of the deformation parameters obtained in the shape model training

stage [11]. In addition, a redundancy checking process can be used for every new generated instance. If it is similar enough to a previous one, it will be deleted.

Problem 2: How to organize the instances to speed up the fetching? We propose to use a tree structure to represent the set of instances. Hierarchical clustering will be used to determine the structure of the resulting index tree.

Problem 3: What kind of shape feature to use? There are a lot of shape features, such as area, circularity, eccentricity, major axis orientation[6], Hu moments[5], etc. In order to keep the pose information and discard the scale and translation parameters, we use the normalized central moments[4] to build the shape feature vector although other features are possible.

Problem 4: What is the appropriate shape similarity metric? In the index tree procedure, the fetched result should be the best one in the instance set to optimize an objective function (E_{area} in Eq. (3)) that quantifies how well the instance matches the potential region group. However, the mapping between distance in feature space and the objective function value may not be monotonic. In our application, we propose to use a coarse to fine, two level searching scheme. In the coarse level, the Euclidean distance of shape features is used as distance measure. In the fine level, a neural network (NN) is used to compute the distance measure. Using a coarse to fine approach yields a reduction of computation and tends to preserve accuracy.

5.1. Sampling in the Deformation Space

In the model training stage, we obtain the distribution of the deformation parameters for the object class. Based on this information, we uniformly sample in the bounded deformation parameter space. We then compute the seven normalized central moments for each generated instance.

In order to decide the size of the instance set such that it has enough instances and little redundancy, we conducted a series of experiments where the size of instance set varied from 500 to 8000. For each example in the training set, we tested all the members of the instance set, and got the best fitting cost by the brute-force method.

The average of the best fitting costs for all the training examples becomes an index for the fitting capability of the instance set. In Fig. 2, we show the average cost values, maximum cost values, and minimum cost values for training examples while the instance set size varies. This experiment was conducted with a leaf model[11], and the size of the example set was about 100. We also conducted experiments for a fish model and a blood cell model, and the results were similar. Based on these experiments, we select the instance set size to be 2000.

5.2. Clustering and Building the Index Tree

To speed up search, we organize the instances in a tree such that the retrieval time can be logarithmic to the number of instances. We use a hierarchical clustering method (minimum variance) to process the shape features

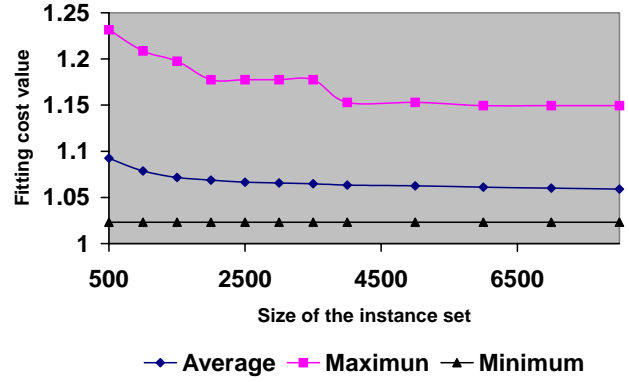


Figure 2: Comparison of the fitting capability

of the instances, and get the tree structure[7]. Although we uniformly sample in the deformation space, there is a hierarchical structure in the corresponding shape feature space.

We used the cophenetic correlation coefficient (CPCC) [7] to validate clustering. The data shows that the generated instance set indeed has a hierarchical structure in the shape feature space.

5.3. Linear Discriminant Function

In searching for the best match based on the index tree structure, the searching time is reduced but it does not guarantee that the nearest match is always found. We tried to use the mean feature of instances in each non-leaf node to select a branch and go to the next level. However, the covariance and distribution for the instances in each node are not the same; furthermore, their distributions are not Gaussian in general.

In order to overcome this problem, we use linear discriminant functions[1] at the non-leaf nodes. Via computing the linear discriminant value, a branch is selected in each non-leaf node. Our experiments verified that this method can increase the success rate in finding the nearest neighbors.

In addition, we propose the following improvements in using the linear discriminant functions. First, if the function shows that the children of the non-leaf node are separable by the solution vector, then we do nothing. Otherwise, we will adjust the instances in the children according to the discriminant values, i.e., move the misclassified instances to the right side.

In addition, we build a small neighbor subset for each discriminant function in each non-leaf node. While using the discriminant function for a shape feature vector, we can get a by-product which is the distance from the point (corresponding to the shape feature vector) to the deciding plane (corresponding to the linear discriminant function). Therefore, for each deciding plane in each node, we can extract a neighbor subset of the instances in the node, which only includes the first n instances nearest to the deciding plane. This work can be done off-line.

In on-line search for a new query vector (a point in the

feature space), for each non-leaf node, if the distance to a deciding plane from the input point is the smallest up to the present, then remember this deciding plane. After a leaf node is reached, we combine the instances of the leaf node with those in the smallest neighbor subset of the deciding plane that has the minimum distance to the input point. The combined set is used in the further processing.

With these improvements, the success rate of finding the nearest neighbors in the leaf node is improved from 75.5% (based on the mean shape feature of the node) to 89.8% (based on the linear discriminant function) for the training examples of the leaf model.

5.4. Neural Network Computation at the Leaf Nodes.

One problem with using the index tree search is that the retrieved result is the nearest neighbor in the shape feature space. However, the distance metric in shape feature space is not the same as the fitting cost (Eq. 1) nor is it monotonic to the fitting cost.

A neural network (NN) can be used to get a mapping from the difference in the shape feature space to the fitting cost measure. We use a three layer back-propagation network with bias terms and momentum[15]. We only use the neural network for the mapping in the leaf nodes to reduce the on-line computation. Because the instances in the same leaf node are more similar, the convergence of the NN training is fast. Also, the accuracy is improved by using different weight sets of NN for different leaf nodes.

In summary, there are two levels of comparison in the retrieval of the similar instance. The shape feature vector is used in the first level (coarse level), and the NN mapping from shape feature difference to fitting cost is used in the second level (fine level).

Based on the index tree searching method, we conducted model fitting experiments for the examples in the training images (about one hundred examples), and the average fitting cost (Eq. 1) was 1.0710. This was very near the average fitting cost by brute force method of matching with each member of the instance set, which is 1.0686. It verified that the index tree searching method has high success rate. We got similar results for the experiments with fish model and blood cell model.

6. Experimental Results

We conducted experiments for deformable object detection and compared the time requirement before and after the improvement in the model fitting. There are hundreds of tested images, including leaf images, fish images, and blood cell images. To obtain results comparable with those using down-hill simplex method, the average speedup is over five times using the index tree-based model fitting method. Fig. 3 shows the object detection for some leaf images using the new model fitting method. Table 1 lists the time requirements for processing the images in Fig. 3 in the region grouping stage of the system. The time unit used is CPU seconds on an SGI O2 workstation.

We also conducted experiments for a database of blood cell micrographs. The database includes about one hundred images. There are many problems with cell image segmentation due to cell attachments, morphological variation, occlusion, presence of faults, artifacts, etc. Our system is able to detect the cells and recover the shape models correctly despite shape and color variation in the cells, and despite the fact that some cells clutter together. Using the index tree-based model fitting method, it takes approximately one order of magnitude less time to finish the processing of images in the database. Some segmentation and model recovery results are shown in Fig. 4.

6.1. Population-based Image Queries

After processing the images in the database of blood cells, we obtain the recovered shape models of detected cells in each image. Based on this information, a population-based image query system can be implemented to satisfy shape retrieval of special interest.

There is a deformation parameter vector corresponding to each recovered shape model. The statistical analysis of shape information is based on these parameter vectors. According to user's interest, some components of this vector can be disregarded in the statistical analysis. For example, in our experiments, we discard the components for translation, rotation, and scale, and we use three components corresponding to stretching, shearing, and bending.

We build a database of blood cell micrographs including about one hundred images. After the object detection for images in the database using our algorithm, we compute a histogram of the shape deformation parameters for each cell image. Via comparing histogram similarity, population based image query is conducted for the database.

In our experiments, there are six bins for each component. We tested and compared retrieval performance using three different histogram similarity metrics: histogram intersection, Chi-squared statistic, and Bhattacharyya distance[12]. The results using these three different metrics was similar. Fig. 5 shows some query results using Bhattacharyya metric. Preliminary experiments indicate that the approach can be used to retrieve cell images with similar shape populations.

7. Conclusion

We presented an index tree-based model fitting procedure that uses pre-computation to accelerate the speed of on-line model fitting, while accuracy of shape-based indexing is maintained. Our system can recover a shape description for each model detected, and allows query formulation based on population of shapes in each image.

Acknowledgments

This work was supported in part through ONR Young Investigator Award N00014-96-1-0661, and NSF grants IIS-9624168 and EIA-9623865.

	(1)	(2)	(3)
The old method	936	2081	1836
The new method	76	186	216

Table 1: Comparison of time requirement in region grouping

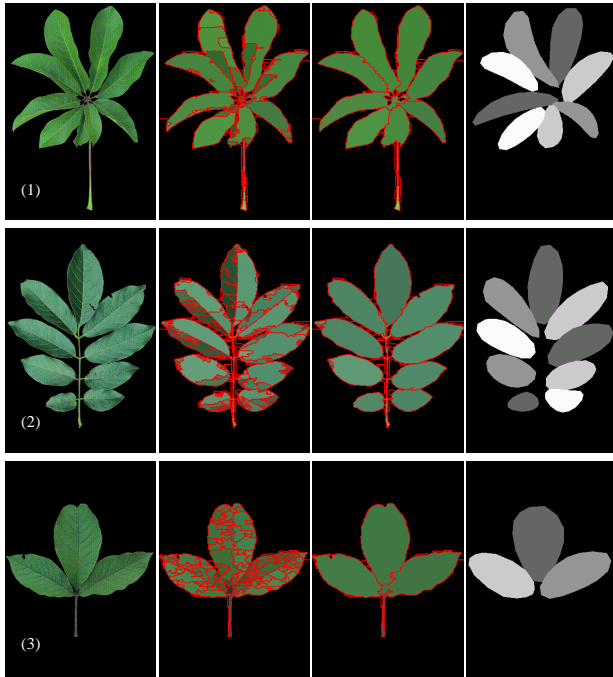


Figure 3: Deformable model-based segmentation and detection for leaf images using the new model fitting method. Each row shows: the original image, the over-segmentation result, the segmentation result after model-based region grouping, and finally the recovered models for detected objects.

References

- [1] R. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
- [2] A. Gersho and R. M. Gray. *Vector quantization and signal compress*. Kluwer, 1992.
- [3] W. Grimson and T. L. Perez. Localizing overlapping parts by searching the interpretation tree. *PAMI*, 9(4):469–482, 1987.
- [4] R. M. Haralick and L. Shapiro. *Computer and Robot Vision*. Addison-Wesley, 1992.
- [5] M. K. Hu. Visual pattern recognition by moments invariants. *IRE Trans. Inform Theory*, (IT-8), 1962.
- [6] A. K. Jain. *Fundamentals of digital image processing*. Prentice-Hall, 1989.
- [7] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice Hall, 1988.
- [8] D. Jelinek and C. J. Taylor. Object recognition from a small number of reference images. *TR MS-CIS-00-02, Grasp laboratory, U. Pennsylvania.*, 2000.
- [9] Y. Lamdan and H. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. *ICCV'88*, pages 238–249.

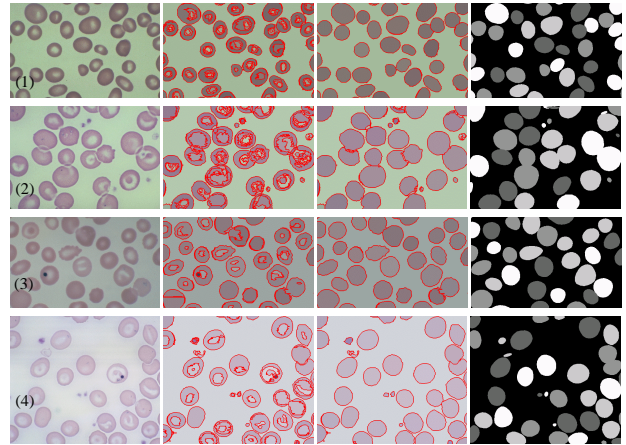


Figure 4: Image segmentation and model recovery for blood cell micrographs based on the new method. Each row shows: the original image, the over-segmentation, the result after model-based region grouping, and finally the recovered models for detected objects.

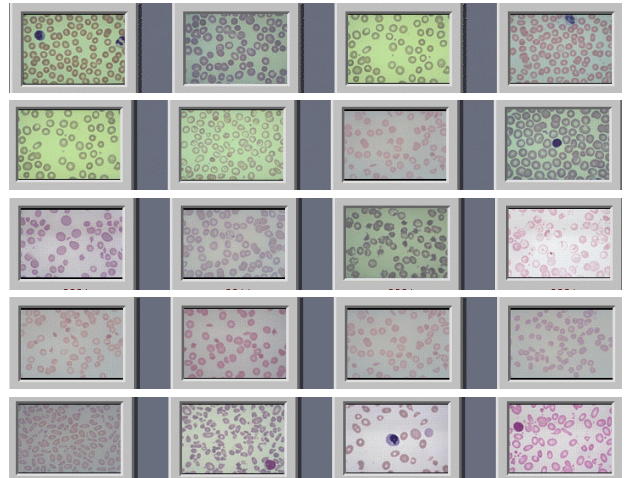


Figure 5: Population-based image query results for the database of blood cell micrographs. In each row, the first is query image, followed by the first three images retrieved by the system as “most similar” from the database.

- [10] A. Leonardis, A. Gupta, and R. Bajcsy. Segmentation as the search for the best description of images in terms of primitives. *ICCV'90*.
- [11] L. Liu and S. Sclaroff. Deformable shape detection and description via model-based region grouping. *CVPR99*, 2:21–27.
- [12] B. Moghaddam, H. Biermann, and D. Margaritis. Defining image content with multiple regions-of-interest. *CBAIVL'99*.
- [13] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *IJCV*, 14(1):5–24, 1995.
- [14] A. P. Pentland. Automatic extraction of deformable part models. *IJCV*, 4(2):107–126, 1990.
- [15] D. Rumelhart, G. Hinton, and R. Williams. *Learning Internal Representations by Error Propagation*. Parallel Distributed Processing, Vol. 1, MIT Press, 1986.