Shape contexts enable efficient retrieval of similar shapes

Greg Mori[†], Serge Belongie[‡] and Jitendra Malik[†]
[†]University of California, Berkeley – Berkeley, CA 94720
[‡]University of California, San Diego – La Jolla, CA 92093
{mori,malik}@cs.berkeley.edu, sjb@cs.ucsd.edu

Abstract

In this work we demonstrate that a recently introduced shape descriptor, the "shape context", can be used to quickly prune a search for similar shapes. Our representation for a shape is a discrete set of n points sampled from its internal and external contours. For each of these points, the shape context is a histogram of the relative positions of the $n-1$ remaining points. We present two methods for rapid shape retrieval: one that does comparisons based on a small number of shape contexts and another that uses vector quantization in the space of shape contexts. We verify the discriminative power of these methods with tests on the Columbia (COIL-100) 3D object database and the Snodgrass and Vanderwart line drawings. The shape context-based methods are shown to quickly produce an accurate shortlist of candidates suitable for a more exact matching engine in spite of pose variation and occlusion.

1 Introduction

We are interested in the use of shape for recognizing 3D objects, represented by a collection of multiple 2D views. A satisfactory theory of shape representation would have a number of desirable attributes:

1. It should support recognition based on exquisitely fine differences e.g. distinguishing faces of twins.
2. At the same time, it should support making coarse discriminations very quickly. Thorpe, Fize and Merlot [22] showed that people, when presented with an image, can answer coarse queries such as presence or absence of an animal in as little as 150ms.
3. The approach should scale to deal with a large number of objects. Biederman[3] has argued that humans can distinguish on the order of 30000 different objects.
4. It should be possible to acquire a representation of an object category from relatively few examples i.e. there should be a good generalization ability.

In this paper we develop further an approach based on the representation of *shape contexts*, introduced in Belongie, Malik and Puzicha [2], which arguably satisfies criteria (1), (2) and (4) above while (3) is yet only a distant possibility[1].

The basic idea of shape contexts is illustrated in Fig. 1. A shape is represented by a discrete set of points sampled from the internal or external contours on the shape. These can be obtained as locations of edge pixels as found by an edge detector, giving us a set $\mathcal{P} = \{p_1, \ldots, p_n\}$, $p_i \in \mathbb{R}^2$, of n points[2]. Consider the set of vectors originating from a point to all other sample points on a shape. These $n-1$ vectors express the configuration of the entire shape relative to the reference point. One way to capture this information is as the *distribution* of the relative positions of the remaining $n-1$ points in a spatial histogram. Concretely, for a point p_i on the shape, compute a coarse histogram h_i of the relative coordinates of the remaining $n-1$ points,

$$h_i(k) = \#\{q \neq p_i \ : \ (q - p_i) \in \text{bin}(k)\} \ .$$

This histogram is defined to be the *shape context* of p_i. We use bins that are uniform in log-polar space, making the descriptor more sensitive to positions of nearby sample points than to those of points farther away. All radial distances are first normalized by the mean distance α between the n^2 point pairs in the shape, thus ensuring that the shape context of a point on a shape is invariant under uniform scaling of the shape as a whole.

As illustrated in Fig. 1, shape contexts will be different for different points on a single shape S; however corresponding (homologous) points on similar shapes S and S' will tend to have similar shape contexts. By construction, the shape context at a given point on a shape is invariant under translation and scaling. Shape contexts are not invariant under arbitrary affine transforms, but the log-polar binning ensures that for small locally affine distortions due to pose change, intra-category variation etc., the change in the shape context is correspondingly small. In addition,

[1]At least it is not provably impossible!
[2]They need not, and typically will not, correspond to key-points such as maxima of curvature or inflection points.

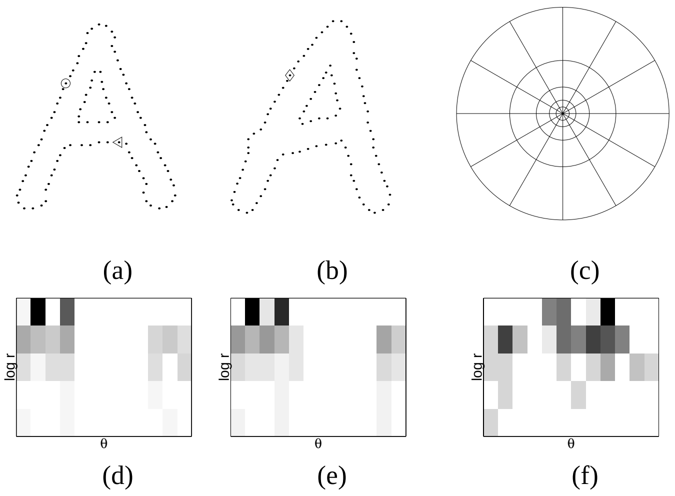

Figure 1: Shape contexts. (a,b) Sampled edge points of two shapes. (c) Diagram of log-polar histogram bins used in computing the shape contexts. We use 5 bins for $\log r$ and 12 bins for θ. (d-f) Example shape contexts for reference samples marked by $\circ, \diamond, \triangleleft$ in (a,b). Each shape context is a log-polar histogram of the coordinates of the rest of the point set measured using the reference point as the origin. (Dark=large value.) Note the visual similarity of the shape contexts for $\circ$ and $\diamond$, which were computed for relatively similar points on the two shapes. By contrast, the shape context for $\triangleleft$ is quite different.

the richness of the shape context descriptor makes it robust to noise and occlusion, as indicated by the experiments reported in [2].

There is a natural way to measure the similarity between two shape contexts. As shown in [2], this facilitates algorithms for solving the correspondence problem between two similar but not identical shapes such as seen in 1(a) and (b). Consider a point p_i on the first shape and a point q_j on the second shape. Let $C_{ij} = C(p_i, q_j)$ denote the cost of matching these two points. As shape contexts are distributions represented as histograms, it is natural to use the χ^2 distance:

$$C_{ij} = \frac{1}{2}\sum_{k=1}^{K}\frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)}$$

where $h_i(k)$ and $h_j(k)$ denote the K-bin normalized histogram at p_i and q_j, respectively. Given the set of costs C_{ij} between all pairs of points i on the first shape and j on the second shape we want to minimize the total cost of matching subject to the constraint that the matching be one-to-one. This is an instance of the square assignment (or weighted bipartite matching) problem, which can be solved in $O(n^3)$ time using the Hungarian method.

We turn now to the use of shape contexts as part of a theory of object recognition based on shape matching. As stated earlier, it is desirable for such a theory to support both accurate fine discrimination, as well as rapid coarse discrimination. This suggests a two stage approach to shape matching, namely:

1. *Fast pruning:* Given an unknown 2D query shape, we should be able to quickly retrieve a small set of likely candidate shapes from a potentially very large collection of stored shapes. The present paper will introduce two algorithms for this problem.

2. *Detailed matching:* Once we have a small set of candidate shapes, we can perform a more expensive and more accurate matching procedure to find the best matching shape to the query shape. An algorithm to achieve this, in a deformable template matching framework, was presented in [2]. This process is computationally expensive (about 200 ms to match two shapes on a 500 MHz Pentium), but very accurate as shown by the experimental results in several domains such as handwritten digit recognition, tests on the Columbia 3D object database [18], and the MPEG-7 shape silhouette database.

The thrust of this paper is in Section 3 where we develop two different algorithms for fast pruning based on shape contexts, resulting in a shortlist of likely candidate shapes to be evaluated later by the more accurate and expensive procedure in [2]. This is preceded by Section 2 on past work and followed by a discussion of scaling to very large collections in Section 4. In Section 5, we show experimental results on the Columbia (COIL-100) 3D object database [19] and the Snodgrass and Vanderwart drawings [21]. We conclude in Section 6.

2 Past Work

An extensive survey of shape matching in computer vision can be found in [23]. Broadly speaking, there are two approaches: (1) feature-based, and (2) brightness-based.

Feature-based approaches involve the use of spatial arrangements of extracted features such as edges or junctions. Silhouettes have been described (and compared) using Fourier descriptors, e.g. [24], skeletons derived using Blum's medial axis transform [20], or directly matched using dynamic programming. Although silhouettes are simple and efficient to compare, they are limited as shape descriptors for general 3D objects because they ignore internal contours and are difficult to extract from real images. Other approaches [10, 9] treat the shape as a set of points in the 2D image, extracted using, say, an edge detector. Lamdan et al. [15] use geometric hashing in a voting scheme. Carlsson

[6] uses *order structure* to compute correspondences. Amit and Geman [1] find key points or landmarks, and recognize objects using the spatial arrangements of point sets. However not all objects have distinguished key points (think of a circle for instance), and using key points alone sacrifices the shape information available in smooth portions of object contours. Other approaches to finding correspondences between points sets include [12] and [7].

Brightness-based approaches make more direct use of pixel brightness values. Several approaches[14, 8] first attempt to find correspondences between the two images, before doing the comparison. This turns out to be quite a challenge as differential optical flow techniques do not cope well with the large distortions that must be handled due to pose/illumination variations. Errors in finding correspondence will cause downstream processing errors in the recognition stage. As an alternative, there are a number of methods that build classifiers without explicitly finding correspondences. In such approaches, one relies on a learning algorithm having enough examples to acquire the appropriate invariances. Some examples include [16, 5] for handwritten digit recognition, [17] for face recognition, and isolated 3D object recognition [18].

3 Fast Pruning using Shape Contexts

Given a large set of known shapes the problem is to determine which of these shapes is most similar to a query shape. From this set of shapes, we wish to quickly construct a short list of candidate shapes which includes the best matching shape. After completing this coarse comparison step one can then apply a more time consuming, and more accurate, comparison technique to only the shortlist. We leverage the descriptive power of shape contexts towards this goal of quick pruning.

We propose two matching methods that address these issues. In the first method, *representative shape contexts*, we compute a few shape contexts for the query shape and attempt to match using only those. The second method, *shapemes*, uses vector quantization to reduce the complexity of the shape contexts from 60-dimensional histograms to quantized classes of shape pieces.

A key component to both of these methods is the solving of nearest neighbour problems. We will denote by $\tau(n_p, n_d)$ the time required to solve a nearest neighbour problem with n_p points in a n_d-dimensional space.

3.1 Representative Shape Contexts

Given two easily discriminable shapes, such as the outlines of a fish and a bicycle, we do not need to compare every pair of shape contexts on the objects to know that they are

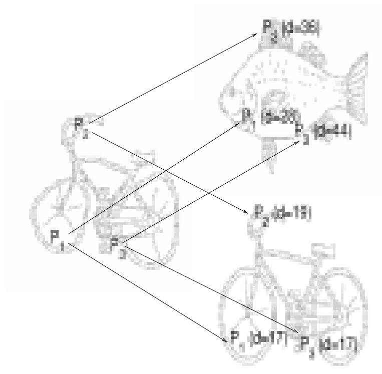

Figure 2: Matching individual shape contexts. Three points on the query shape (left) are connected via arrows to their best matches on two known shapes. χ^2 distances are given with each matching.

different. When trying to match the dissimilar fish and bicycle, none of the shape contexts from the bicycle have good matches on the fish – it is immediately obvious that they are different shapes. Figure 2 demonstrates this process.

In concrete terms, the matching process proceeds in the following manner. For each of the known shapes S_i, we precompute a large number s (about 100) of shape contexts $\{SC_i^j : j = 1, 2, \ldots, s\}$. But for the query shape, we only compute a small number r ($r = 5$ in experiments) of shape contexts. To compute these r shape contexts we randomly select r sample points from the shape. We use all the sample points on the shape to fill the histogram bins for the shape contexts corresponding to these r points. We then do comparisons with each of the known shapes using only these shape contexts.

To compute the distance between a query shape and a known shape, we find the best matches for each of the r shape contexts. This involves performing r nearest-neighbour searches. The distance to a known view is defined to be the sum of these r distances. Distances are computed using the χ^2 distance.

$$dist(S_{query}, S_i) = \sum_{j=1}^{r} \chi^2(SC_{query}^j, SC_i^*)$$

where $SC_i^* = argmin_u \chi^2(SC_{query}^j, SC_i^u)$

We then find the closest matches by comparing these distances.

Pseudocode for Representative Shape Context method:

```
PRE-PROCESSING:
    % Compute shape contexts for known shapes

PRUNING:
    $SC_{query} =$ shape contexts for $r$ random points
    foreach known shape $S_i$
        for $j = 1 : r$
            $dist(S_{query}, S_i) += \min_u(\chi^2(SC^j_{query}, SC^u_i))$
    % Sort $dist$ and truncate to return a
    % shortlist.
```

The pruning phase requires $O(rN \cdot \tau(s,d))$ time, where N is the number of known shape views.

3.2 Shapemes

The second matching method uses vector quantization on the shape contexts. The full set of shape contexts for the known shapes consists of $N \cdot s$ d-dim vectors. A standard technique in compression for dealing with such a large amount of data is vector quantization. Vector quantization involves clustering the vectors and then representing each vector by the index of the cluster that it belongs to. We call these clusters *shapemes* – canonical shape pieces. Figure 3 shows the representation of sample points as shapeme labels.

To derive these shapemes, once again all of the shape contexts from the known set are considered as points in a d-dimensional space. We do k-means clustering to obtain k shapemes.

We represent each known view as a collection of shapemes. Each d bin shape context is quantized to its nearest shapeme, and replaced by the shapeme label (an integer in $\{1, \ldots, k\}$). A known view is then simplified into a histogram of shapeme frequencies. No spatial information amongst the shapemes is stored. We have reduced each collection of s shape contexts (d bin histograms) to a single histogram with k bins.

In order to match a query shape, we simply perform this same vector quantization and histogram creation operation on the shape contexts from the query shape. We then find nearest neighbours in the space of histograms of shapemes.

Pseudocode for the Shapeme method:

```
% Vector Quantize and Bin:  Replace each
% shape context by closest cluster center.
% Compute frequencies of centers.
VQandBin($Shapemes$, $SC$)
    $ShapemeCounts$ = zeros(k,1)
    foreach shape context $SC^j$
        $c = argmin_i \chi^2(SC^j, Shapemes_i)$
        $ShapemeCounts(c)++$
    return $ShapemeCounts$

PRE-PROCESSING:
    $SC_{all} =$ Shape contexts of all known shapes
    % KMEANS clusters the vectors in $SC_{all}$
    % into $k$ clusters, returns the centers of
    % those clusters.
    $Shapemes = $ KMEANS($SC_{all}, k$)
    foreach known shape $S_i$
        % $ShCounts_i$ is a histogram of shapeme
        % counts.
        $ShCounts_i = $ VQandBin($Shapemes, SC_i$)

PRUNING:
    $ShCounts_{query} = $ VQandBin($Shapemes, SC_{query}$)
    foreach known shape $S_i$
        $dist(S_{query}, S_i) = \chi^2(ShCounts_{query}, ShCounts_i)$
    % Sort $dist$ and truncate to return a
    % shortlist.
```

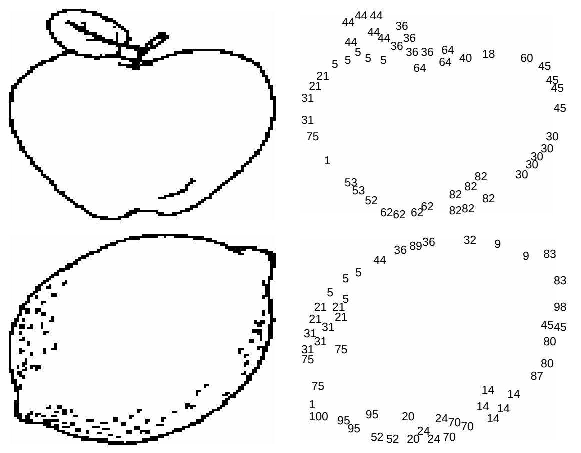
(a) (b)

Figure 3: (a)Line drawing, (b)sampled points with shapeme labels. $k = 100$ shapemes were extracted from a known set of 260 shapes (26000 shape contexts). Note the similarities in shapeme labels (31,21,5 on left side, 45 on right side) between similar portions of the shapes.

The pruning phase of this shapeme-based matching process takes $O(s \cdot \tau(k,d))$ time to do the vector quantization (assigning the query shape's shape contexts to shapemes, collecting these shapeme frequencies into histograms) and $\tau(N,k)$ to do the final search amongst the histograms of shapemes.

4 Scalability

The two matching methods we present require solving nearest-neighbour problems in some D-dimensional space with $O(N)$ points (N is the number of known shapes). The naive algorithm for doing such nearest-neighbour searches costs $\tau(O(N), D) = O(ND)$ time. This brute-force approach is not viable as N becomes large. If we wish to design a system that can handle on the order of 10^4 objects we must reduce this $O(ND)$ complexity.

Recent work in the theory community on the ϵ-*approximate nearest neighbours* (ϵ-NN) problem can be applied here. The ϵ-NN problem is to find a point $p \in P$

that is the ϵ-nearest neighbour of the query point q: for all $p' \in P, d(p,q) \leq (1+\epsilon)d(p',q)$. Indyk and Motwani [11] describe an algorithm for doing ϵ-NN queries in $\tau(N,D) = O(Dpolylog(N))$ time that uses random projections and the Johnson-Lindenstrauss lemma [13]. Using an algorithm of this nature, we could perform our pruning methods efficiently. Moreover, since we are constructing a shortlist, and are not sensitive to small $(1+\epsilon)$ scalings in distance, getting precise results from a nearest neighbour algorithm is not critical.

5 Results

We use the Columbia (COIL-100) 3D object database and the Snodgrass and Vanderwart line drawings as our test sets. In the following subsections we present graphs showing the performance of the two methods on these test sets.

The graphs plot error rate vs. pruning factor (on a $\log$ scale) for various degrees of distortion and occlusion. The error rate computation assumes a perfect detailed matching phase. That is, a query shape produces an error only if there is no correctly matching shape in the shortlist produced by the pruning method. The x-axis on each of the graphs shows the length of the shortlist. Pruning factor is defined to be $N/length(Shortlist)$. For example, with $N = 260$ known shapes, if the pruning factor is 26 then the shortlist has 10 shapes in it.

In general the representative shape contexts method performs better – particularly when dealing with occlusions. Missing a couple of shape contexts won't spoil the matching. Moreover, the shapemes are more easily corrupted by occluded points and distortions. However, the vector quantization used in shapemes does buy us computational speed.

5.1 COIL-100

The first experiment involves the COIL-100 database. The database consists of 100 unique objects. Each object was placed on a turntable and photographed every 5 degrees for a total of 72 views per object. We prepared our sets of known shapes by selecting a number of equally spaced views for each object and using the remaining views for querying. We use a Canny edge detector to extract line features from the images. These edges are then sampled to create point features for use in shape contexts.

We ran experiments using 4, 8, and 12 (corresponding to $90°$, $45°$, and $30°$ spacing) known views per object. Figures 4 and 5 show the results for these tests. Both of the pruning methods are successful: for example, with 12 known views per object a pruning factor of 100 (shortlist of length 12) can be obtained with an error rate of 9% for the representative shape contexts method, and 10% for the shapeme method.

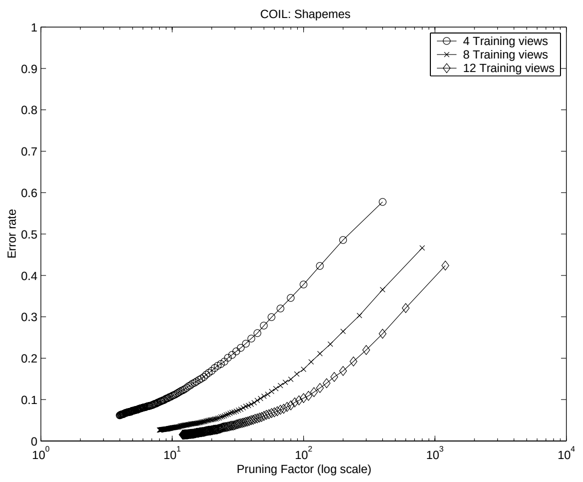

Figure 4: Error rate vs. pruning factor on COIL-100 dataset using shapemes. Pruning factor is defined to be $N/length(Shortlist)$. For example, with $N = 260$ known shapes, if the pruning factor is 26 then the shortlist has 10 shapes in it.

Figure 6 shows some of shortlists on the COIL-100 dataset using the shapeme matching method. Many of the errors on this dataset involve objects that are nearly indistinguishable in terms of shape. For example, the shape matching processes are readily confused by the toy cars of different colour. In addition, there are a few brands of pop and coffee mugs with different patterns on them in the COIL-100 dataset. Relying solely on shape, without cues such as colour and texture, it is difficult to differentiate between the members of these groups of objects.

5.2 Snodgrass & Vanderwart

The second experiment uses the Snodgrass & Vanderwart line drawings [21]. This dataset contains line drawings of 260 commonly occurring objects. They are a standard set of objects that have been frequently used in the psychophysics community for tests with human subjects. The only information available for object recognition in this dataset is shape – this makes it an excellent dataset on which to test our matching methods. Since the images are only line drawings, no preprocessing phase of edge extraction is needed. We just sample points from the line drawings directly.

The Snodgrass & Vanderwart dataset has only one image per object. We use these original images as the known set, and create a synthetic distorted set of images for querying. The thin plate spline (TPS) model, which is commonly used for representing flexible coordinate transformations [4], is used to create these distortions. In a 2D view of a class of 3D object there are two sources of variation: pose change

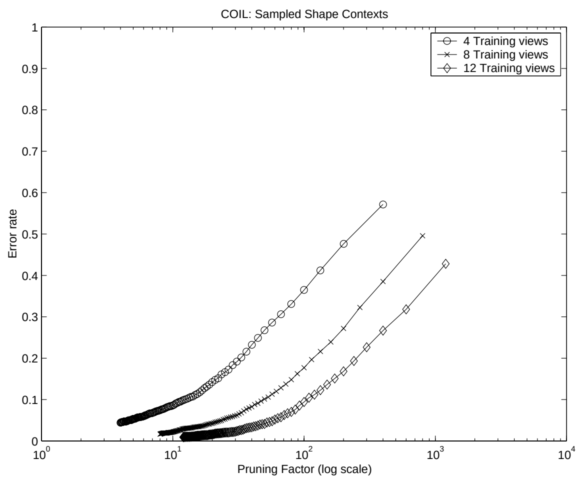

Figure 5: Error rate vs. pruning factor on COIL-100 dataset using representative shape contexts

and intra-class change. We use the non-linear TPS model to simulate both of these types of variation simultaneously. We apply a random TPS warp of fixed bending energy to a reference grid, and use this warp to transform the edge points of a line drawing.

In addition to distortions, we test the ability of our pruning methods to deal with occlusions. We take the set of TPS-distorted objects and subject them to random occlusions. The occlusions are generated using a linear occluding contour. The query objects in Figure 7 show some distorted and occluded Snodgrass & Vanderwart images. Note that the occluding contour is included – we will sample points from it when creating the shape contexts.

The 260 original Snodgrass & Vanderwart images were used as the known set. We generated 5200 distorted images (20 per original image) and 5200 distorted and occluded images for use as query sets. The occluded images were split into levels of difficulty according to the percentage of edge pixels lost under occlusion. Figures 9 and 10 show the results for our two pruning methods. The pruning methods are both very effective in dealing with the TPS-distorted images. The shapeme method can achieve a pruning factor of ≈ 100 (a correct match in a shortlist of length 3 out of 260 images) with an error rate of only 13%, while the representative shape contexts method only has an error rate of 2%.

The power of the representative shape contexts method comes out in the occlusion tests. Even with extremely difficult levels of occlusion (20%-30% and 30%-40%) we can still obtain large amounts of pruning with reasonable error rates.

Figures 7 and 8 show some example shortlists on the Snodgrass & Vanderwart dataset using the representative shape contexts method.

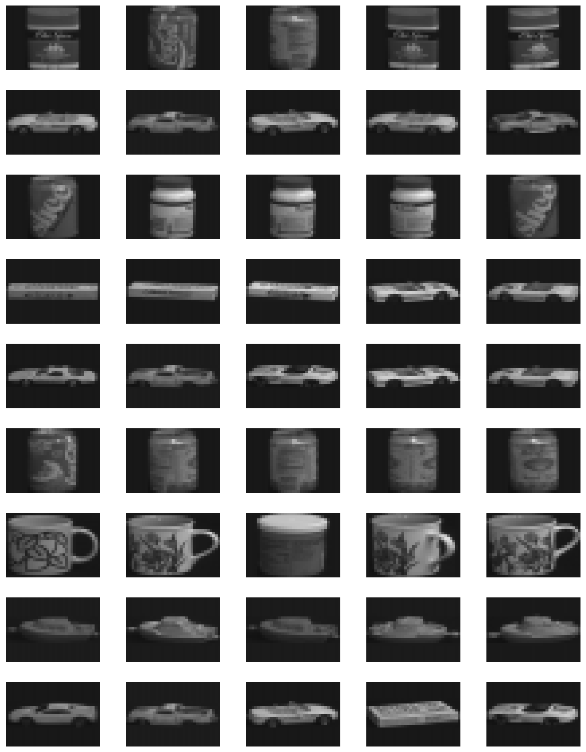

Figure 6: Some shortlists found for COIL-100 images. The first column shows query objects. The remaining columns show the closest 4 matches to each query object using the representative shape contexts matching method. A query is successful if there is at least one matching object on the shortlist.

6 Conclusion

Previous work on shape matching via a deformable template-based framework has been very successful for object recognition. However, these methods are too expensive computationally to be used on a large scale object database. We have shown how a shape context-based pruning approach can assist by constructing an accurate shortlist in order to reduce this computational expense. We proposed two methods of matching – one using a small number of representative shape contexts, and the other based on vector quantization of shape contexts into shapemes. Both methods were shown to perform well as efficient pruning mechanisms on the COIL-100 and Snodgrass & Vanderwart datasets, and deal robustly with occlusion and pose or intra-class variation.

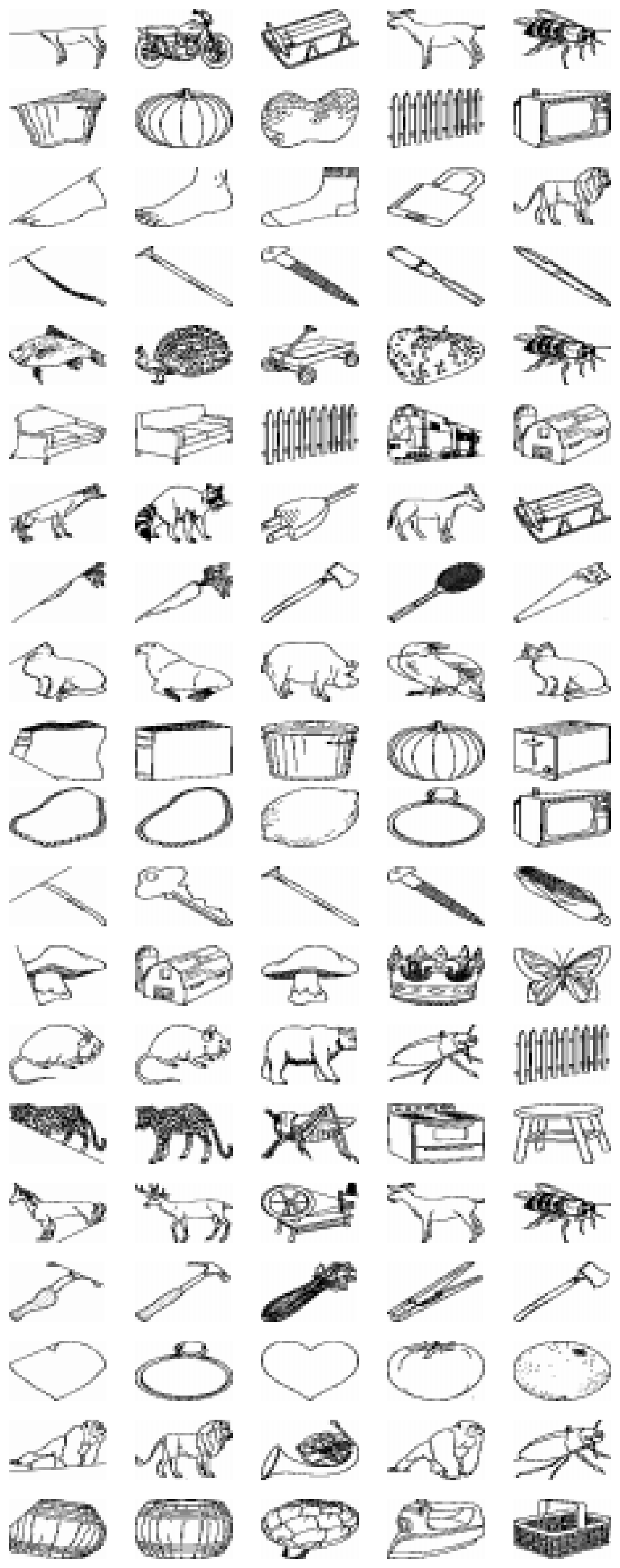

Figure 7: Some shortlists for the distorted and occluded Snodgrass & Vanderwart dataset using the representative shape contexts method. The first column is the query object. Remaining 4 columns show closest matches to each query object. An example of a successful query is the 3^{rd} row. The query object is a foot with the top portion occluded (note the presence of the occluding contour). The first entry in the shortlist is a correct match, followed by a sock (similar shape), and a lock (has a straight edge similar to the occluding contour in the query image). The 2^{nd} row shows a failure. The query object is a partially occluded garbage can. None of the objects on the shortlist correctly match, but they do share some similarity (vertical edges).

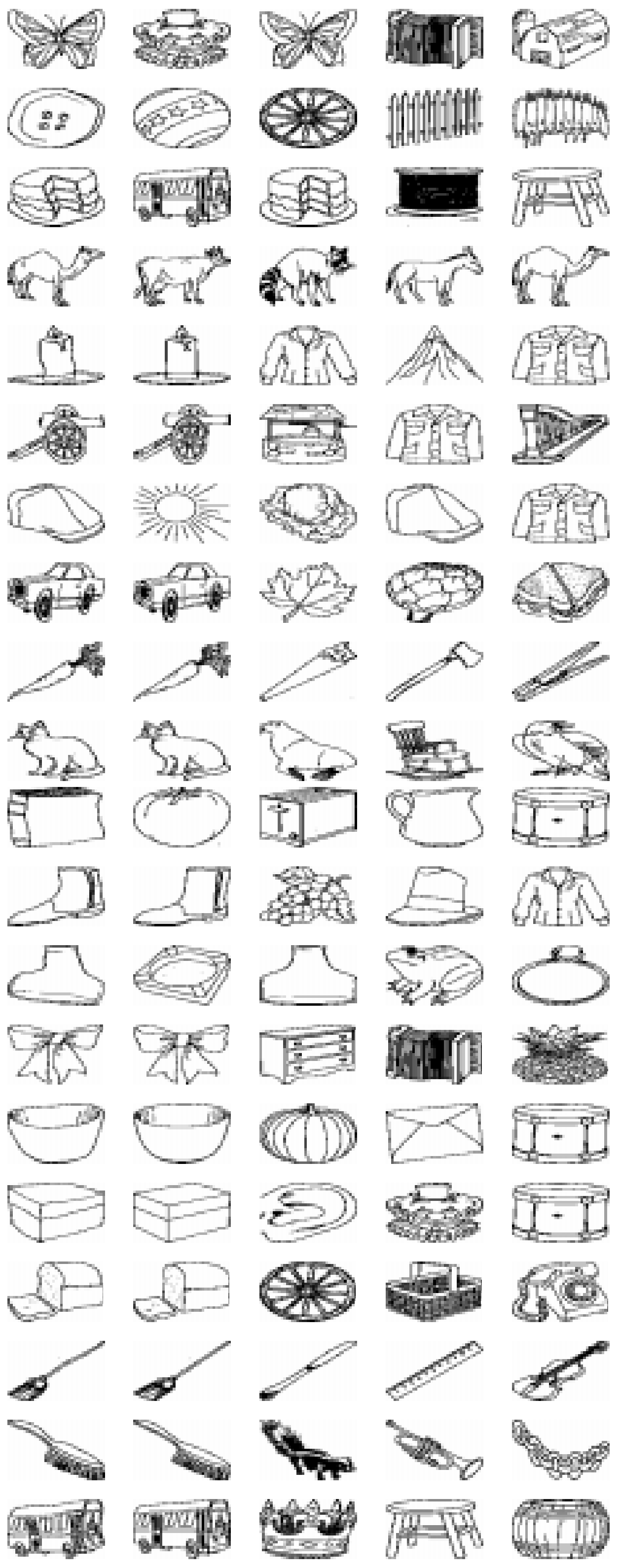

Figure 8: Some shortlists for the distorted Snodgrass & Vanderwart dataset using the representative shape contexts method. The first column is the query object. Remaining 4 columns show closest matches to each query object. The 4^{th} row shows a successful query. The query object is a distorted version of the camel. All 4 objects in the shortlist are similar shapes (animals). The 2^{nd} row shows a failure. The query object is a distorted round button. The shortlist contains other round objects, but no button.

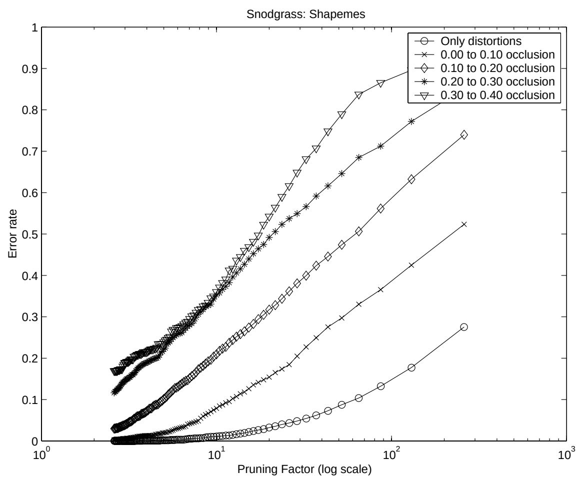

Figure 9: Error rate vs. pruning factor on Snodgrass dataset using shapemes

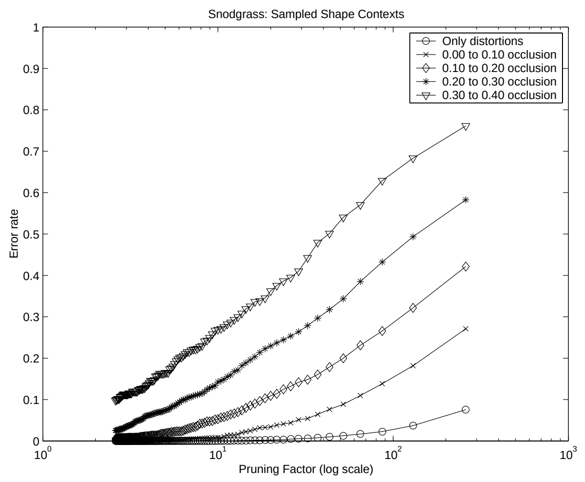

Figure 10: Error rate vs. pruning factor on Snodgrass dataset using representative shape contexts

References

[1] Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers. *IEEE Trans. PAMI*, 19(11):1300–1305, November 1997.

[2] S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *Eighth IEEE International Conference on Computer Vision*, volume 1, pages 454–461, Vancouver, Canada, July 2001.

[3] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[4] F. L. Bookstein. Principal warps: thin-plate splines and decomposition of deformations. *IEEE Trans. PAMI*, 11(6):567–585, June 1989.

[5] C. Burges and B. Schölkopf. Improving the accuracy and speed of support vector machines. In *NIPS*, pages 375–381, 1997.

[6] S. Carlsson. Order structure, correspondence and shape based categories. In *Shape Contour and Grouping in Computer Vision*, pages 58–71. Springer LNCS 1681, 1999.

[7] H. Chui and A. Rangarajan. A new algorithm for non-rigid point matching. In *CVPR*, volume 2, pages 44–51, June 2000.

[8] T. Cootes, D. Cooper, C. Taylor, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, Jan. 1995.

[9] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *Proc. 7th Int. Conf. Computer Vision*, pages 87–93, 1999.

[10] D. Huttenlocher, R. Lilien, and C. Olson. View-based recognition using an eigenspace approximation to the Hausdorff measure. *PAMI*, 21(9):951–955, Sept. 1999.

[11] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *ACM Symposium on Theory of Computing*, pages 604–613, 1998.

[12] A. E. Johnson and M. Hebert. Recognizing objects by matching oriented points. In *CVPR*, pages 684–689, 1997.

[13] W. Johnson and J. Lindenstrauss. Extensions of lipshitz mapping into hilbert space. *Contemp. Math.*, 26:189–206, 1984.

[14] M. Lades, C. Vorbrüggen, J. Buhmann, J. Lange, C. von der Malsburg, R. Wurtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Trans. Computers*, 42(3):300–311, March 1993.

[15] Y. Lamdan, J. Schwartz, and H. Wolfson. Affine invariant model-based object recognition. *IEEE Trans. Robotics and Automation*, 6:578–589, 1990.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[17] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11):1771–1782, November 2000.

[18] H. Murase and S. Nayar. Visual learning and recognition of 3-D objects from appearance. *Int. Journal of Computer Vision*, 14(1):5–24, Jan. 1995.

[19] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100). Technical Report CUCS-006-96, Columbia Univ., 1996.

[20] D. Sharvit, J. Chan, H. Tek, and B. Kimia. Symmetry-based indexing of image databases. *J. Visual Communication and Image Representation*, 1998.

[21] J. G. Snodgrass and M. Vanderwart. A standardized set of 260 pictures: Norms for name agreement, familiarity and visual complexity. *Journal of Experimental Psychology: Human Learning and Memory*, 6:174–215, 1980.

[22] S. Thorpe, D. Fize, and Merlot. Speed of processing in the human visual system. *Nature*, 381:520–522, 1996.

[23] R. C. Veltkamp and M. Hagedoorn. State of the art in shape matching. Technical Report UU-CS-1999-27, Utrecht, 1999.

[24] C. Zahn and R. Roskies. Fourier descriptors for plane closed curves. *IEEE Trans. Computers*, 21(3):269–281, March 1972.