2D Shape Classification and Retrieval

Graham McNeill, Sethu Vijayakumar
Institute of Perception, Action and Behavior
School of Informatics, University of Edinburgh, Edinburgh, UK. EH9 3JZ
Email: G.J.McNeill-2@sms.ed.ac.uk, sethu.vijayakumar@ed.ac.uk

Abstract

We present a novel correspondence-based technique for efficient shape classification and retrieval. Shape boundaries are described by a set of (ad hoc) equally spaced points – avoiding the need to extract "landmark points". By formulating the correspondence problem in terms of a simple generative model, we are able to efficiently compute matches that incorporate scale, translation, rotation and reflection invariance. A hierarchical scheme with likelihood cut-off provides additional speed-up. In contrast to many shape descriptors, the concept of a mean (prototype) shape follows naturally in this setting. This enables model based classification, greatly reducing the cost of the testing phase. Equal spacing of points can be defined in terms of either perimeter distance or radial angle. It is shown that combining the two leads to improved classification/retrieval performance.

1 Introduction

In many object recognition problems, the examples are most easily disambiguated on the basis of *shape* - as opposed to features such as color or texture. Often the classification of objects extracted from an image database are most intuitively formulated as a shape classification task. Here we present a fast, general algorithm for classification and retrieval of 2D objects.

Much of the recent work in this area uses a finite set of points taken from the object's boundary as the *shape representation*. Points can be selected on the basis of maximal curvature [Super, 2004], distance from the centroid [Zhang *et al.*, 2003] or any criteria deemed suitable to the class of shapes involved. More sophisticated approaches parameterize the boundary as a closed curve and slide points along the outline to minimize an objective function (e.g. [Wang *et al.*, 2004]). These methods produce good results but generally use expensive optimization algorithms. An alternative to finding the 'correct points' is to simply place points at roughly equal intervals along the boundary. Belongie et al. [Belongie *et al.*, 2002] used this approach effectively in their work on shape contexts.

Having chosen how to represent the objects, one must solve the *correspondence problem* in order to compare shapes. This is illustrated in Fig.1. Also, a valid set of transformations for shape matching is required. These transformations represent the chosen definition of shape, i.e. they are the operations that leave shape unchanged.

In this paper, we introduce a simple, generic technique for shape classification and retrieval. Shapes are represented by a potentially large number of points from their boundaries. The points lie at fixed intervals in terms of distance along the boundary or radial angle. This gives an accurate and robust description of shape that avoids the somewhat arbitrary decision of what constitutes a good point. The high computational cost associated with using many points is reduced in three ways. Firstly, a hierarchical approach avoids the need to consider all possible label assignments. Secondly, potential correspondences are considered simultaneously and a clear winner often emerges early in the computation. Finally, for classification problems, the algorithm produces class means with associated variances at each point. This allows model based classification, requiring fewer shape comparisons at the testing stage. By considering both the distance and angle based descriptors, we can accurately describe a wider range of shapes and increase the classification/retrieval accuracy. The effectiveness of the method is demonstrated on a benchmark data set and compared to other state-of-the-art methods in the field.

2 Method

We wish to develop generic techniques for classifying or retrieving shapes. There are two important issues to consider: Firstly, the ultimate goal is to work with large data sets and hence, computational expense is an important factor. Secondly, the shapes involved may be very diverse, with large differences between some classes. For example, one class might consist of very jagged shapes and the other smoothly curved shapes. This implies that overly specific models of shape variation should be avoided.

2.1 Shape Correspondence

Given two shapes, we wish to optimally align them in order to observe the genuine differences in shape. When shapes are represented by a set of points, it is necessary to find the best match over all possible label assignments (Fig.1). In this paper, we use a definition of shape from statistical shape analysis (e.g. [Dryden and Mardia, 1998]): The *shape* of a set of labelled points is all the information that is invariant to rotation, scaling and translation. In addition to this, we also consider reflection to be a valid transformation. Allowing

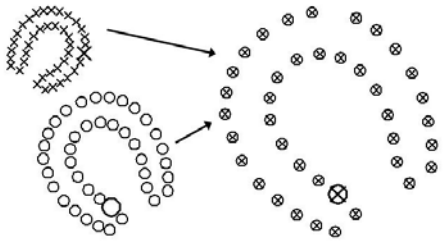

Figure 1: Two identical shapes (left). Large symbols indicate corresponding points under a random assignment of labels. To align the shapes, we must find the *correct labels* as well as the *correct transformations* (right).

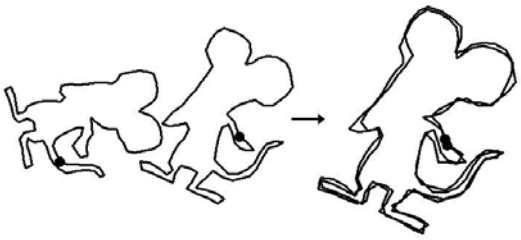

Figure 2: Two shapes from the same class, each described by 100 points (left) - linear interpolation is used to aid visualization. The correspondence algorithm correctly cycles the labels and the shapes can be aligned (right).

only basic transformations reflects our lack of prior knowledge about the shape classes involved. We now formulate the correspondence problem in a probabilistic fashion. This will enable correspondences to be chosen at minimal computational cost.

Correspondence Algorithm

Consider two shapes, each represented by k points from their respective boundaries: $\mathbf{X} = (\mathbf{x}_1 \dots \mathbf{x}_k)^T$ and $\mathbf{Y} = (\mathbf{y}_1 \dots \mathbf{y}_k)^T \in \mathbb{R}^{k\times 2}$. Both shapes are *centered*: $\sum_{j=1}^k \mathbf{x}_j = \sum_{j=1}^k \mathbf{y}_j = \mathbf{0}$. We assume that the $\mathbf{y}_j$ are generated independently from

$$\mathbf{y}_j \sim N(s\Gamma\mathbf{x}_{j+m|k} + \mathbf{t}, \sigma^2\mathbf{I}_2) \qquad (1)$$

for $j = 1, \dots, k$, where s is a scaling parameter, $\mathbf{t}$ a translation vector and Γ a 2D rotation matrix - reflection is dealt with later. $m \in \{0, \dots, k-1\}$ cycles the labels of the points in X. Finding the maximum likelihood estimates (MLEs) of the parameters is a restricted version of the matching problem for two unlabelled sets investigated in [Kent *et al.*, 2004]. For the general case, the large number of potential correspondences ($k!$) can make finding the MLEs prohibitively expensive. However, here we are dealing with boundaries and need only consider cycling of the labels. This means that there are only k potential label assignments and the computational cost of computing the MLEs $\hat{\Gamma}, \hat{s}, \hat{\mathbf{t}}$ and $\hat{m}$ is more acceptable.

The likelihood of the parameters can be written as

$$L(m, s, \mathbf{t}, \Gamma) = \frac{1}{(2\pi\sigma^2)^k} \exp\{\frac{-1}{2\sigma^2} d^2(\mathbf{Y}, \mathbf{X})\} \qquad (2)$$

where

$$d^2(\mathbf{Y}, \mathbf{X}) = \sum_{j=1}^{k} \|\mathbf{y}_j - (s\Gamma\mathbf{x}_{j+m|k} + \mathbf{t})\|^2.$$

For a fixed $m \in \{0, ..., k-1\}$, $\hat{s}, \hat{\mathbf{t}}$ and $\hat{\Gamma}$ take values that minimize $d^2(\mathbf{Y}, \mathbf{X})$ - a consequence of assuming isotropic variance and independence between points. We now exploit a closed form solution for the minimum of $d^2(\mathbf{Y}, \mathbf{X})$ which can be written compactly using complex notation. Letting $\mathbf{x}_j \equiv (x_{j1}, x_{j2})^T \rightarrow x_{j1} + ix_{j2} \equiv z_j \in \mathbb{C}$, the shapes become complex vectors: $\mathbf{X} \rightarrow \mathbf{z}, \mathbf{Y} \rightarrow \mathbf{w} \in \mathbb{C}^k$. Following [Dryden and Mardia, 1998] we have, for a fixed cycling m, and unit sized shapes ($\sum_{j=1}^k |z_j|^2 = \sum_{j=1}^k |w_j|^2 = 1$)

$$\min_{\Gamma, s, \mathbf{t}} d^2(\mathbf{X}, \mathbf{Y}) = 1 - |\mathbf{w}^*\mathbf{z}|^2$$

where $\mathbf{w}^*$ denotes the conjugate transpose of $\mathbf{w}$. Absorbing the size normalization into the distance expression gives the expression for the *full Procrustes distance*:

$$d_F(\mathbf{z}, \mathbf{w}) = 1 - \frac{|\mathbf{w}^*\mathbf{z}|^2}{\mathbf{w}^*\mathbf{w}\mathbf{z}^*\mathbf{z}}. \qquad (3)$$

This is a standard metric in shape analysis. Substituting $d_F^2(\mathbf{z}, \mathbf{w})$ into eq.(2) allows us to evaluate $L(m, \hat{s}, \hat{\mathbf{t}}, \hat{\Gamma})$ for all $m \in \{0, ..., k-1\}$ and hence, find $\hat{m}$. Fig.1 demonstrates the correspondence algorithm applied to two identical shapes - alignment is achieved using the full Procrustes fit (Section 2.2). As expected, the shapes can be perfectly matched. In Fig.2, the shapes are different examples from the same class. A good match is possible after finding the best correspondence.

Invariance with respect to reflection is not yet incorporated into the distance computation. It is worth mentioning that Procrustes distances can directly incorporate reflection invariance [Mardia *et al.*, 1979]. Unfortunately, this idea is not compatible with our formulation of the correspondence problem. A simple example demonstrates this: Imagine $\mathbf{z}$ is the reflection of $\mathbf{w}$. When faced with the task of corresponding the two shapes, we are obviously unaware of the reflection and proceed to label both shapes in an anticlockwise direction. $\mathbf{w}$ will only be the same as its reflection (in the Procrustes sense) if the labels are reflected along with the points. Doing this would flip the direction of the labels to clockwise. $\mathbf{z}$ cannot be matched to the original $\mathbf{w}$ (due to reflection) nor to the reflected $\mathbf{w}$ (because the labels run in opposite directions). The reflection of $\mathbf{z}$ is given by $\mathbf{z}^* \equiv (\bar{z_1}, \dots, \bar{z_k})$. This operation changes the direction of the labels to clockwise. Reversing the order of the entries: $\mathbf{z}_R \equiv (\bar{z_k}, \dots, \bar{z_1})^T$, switches the direction back to anticlockwise. The optimal correspondence between $\mathbf{z}_R$ and $\mathbf{w}$ can now be found as normal. This apparent doubling in computational expense can be significantly reduced by tracking the correspondence calculations for the non-reflected shapes.

Reducing the Computational Cost

Any classification or retrieval problem will require a large number of correspondences to be found. The cost of computing correspondences is dominated by k - the number of

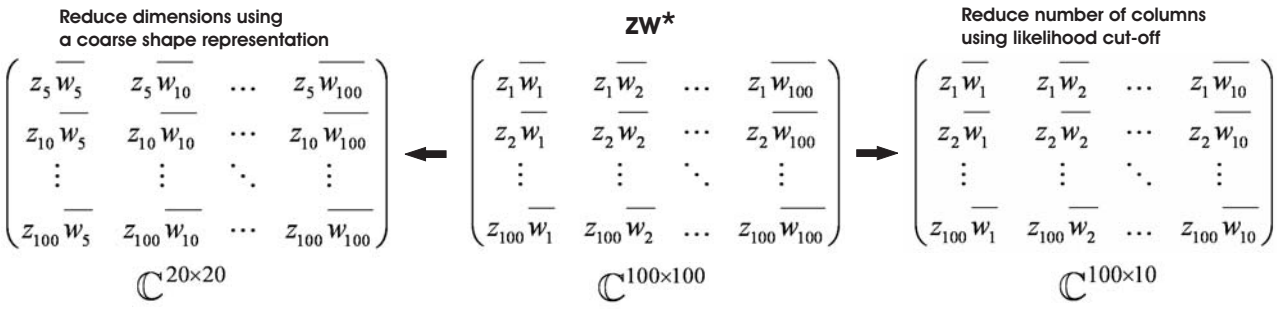

Figure 3: Example of how the speed-up techniques can affect the dimensions of the outer product matrix - see text for details.

boundary points used. Firstly, the number of cycles to consider increase linearly with k. Secondly, the distance computation used to evaluate a correspondence is essentially a dot product of vectors in $\mathbb{C}^k$. From an accuracy perspective, it may be necessary to have a large k. Fortunately, for any given k, there are options for reducing both the number of cycles that are considered and the cost of each distance computation.

To evaluate $d_F^2(\mathbf{z}, \mathbf{w})$ it is necessary to compute the dot products $\mathbf{w}^*\mathbf{w}$, $\mathbf{z}^*\mathbf{z}$ and $\mathbf{w}^*\mathbf{z}$. The values of $\mathbf{w}^*\mathbf{w}$ and $\mathbf{z}^*\mathbf{z}$ do not change with the cycle value and need only be computed once. The cost of corresponding $\mathbf{z}$ and $\mathbf{w}$ is therefore dominated by calculating the dot product between $\mathbf{w}^*$ and all cycled versions of $\mathbf{z}$. This is equivalent to summing along the diagonals of the outer product matrix $\mathbf{z}\mathbf{w}^*$ (for example, summing the leading diagonal gives the value of $\mathbf{w}^*\mathbf{z}$ when the cycle value is zero - see Fig.3). The following procedures reduce the dimensions of $\mathbf{z}\mathbf{w}^*$ and hence, the computational cost of the algorithm.

The number of points used, k, is assumed to be large (say $k = 100$). To avoid considering all k possible values of the cycling parameter m, we initially consider a coarse representation of the shapes using $k' < k$ points (say $k' = 20$ and the points are found by selecting every fifth point of the 100). The likelihoods for all 20 values of the cycling parameter m' are then evaluated as normal. It seems reasonable to assume that the most likely m will correspond the shapes in a similar way to that which would be achieved using all 100 points. Since the 20 points come from the original 100, it is easy to relate these approximate correspondences back to the true shapes (those described by 100 points). The final correspondences are found in the standard way, but only cycle values close to the approximated value are investigated. The outer product matrix is in $\mathbb{C}^{20\times20}$ rather than $\mathbb{C}^{100\times100}$ for the first round of correspondences (Fig.3). At the fine tuning stage, only a small number of the potential 100 correspondences are evaluated. Thus, the overall cost is greatly reduced.

Regardless of the number of points used to describe a shape, it seems that the full matrix $\mathbf{z}\mathbf{w}^*$ must be evaluated and used to compute the Procrustes distances. However, by computing the columns of $\mathbf{z}\mathbf{w}^*$ incrementally, the likelihoods over all k possible cycles can be monitored and a correspondence chosen when one of them exceeds a given threshold (c.f. Fig.3). This approach captures the intuitive idea that some correspondences will be much easier to find than others. The likelihood threshold is reached after fewer iterations when 'new' points are selected at random (e.g. $\bar{w_{137}}\mathbf{z}, \bar{w_{49}}\mathbf{z}, \dots$) so that the full boundary is explored quickly.[1] Clearly if the threshold is never reached and all the points are needed, this incremental method will be more costly than the basic algorithm. The additional cost comes from updating the k different $\mathbf{z}^*\mathbf{z}$ associated with each labelling of $\mathbf{z}$ and from evaluating the likelihood at each stage.[2] It can be partly avoided by evaluating $\mathbf{z}\mathbf{w}^*$ in blocks, rather than single columns. Often very few points are required to find the best correspondence (Section 3.3). It is worth noting that our algorithm could assign correspondences on the basis of minimum Procrustes distance (d_F) - so why bother with likelihoods? Fig.4 shows the evolution of the likelihoods compared to the squared full Procrustes distances for a typical problem. The likelihood approach magnifies the best correspondences and kills-off the worst, leading to a quicker choice of cycle parameter m. Recall from eq.(1) that there is isotropic variance, σ^2, at each point. σ^2 is a free parameter that affects how quickly the likelihoods peak to a single cycle value. A small variance encourages fast correspondence selection (Fig.4 (right)) but assumes little within class variability. Note that this thresholding technique is applicable whenever the correspondence algorithm is used, i.e. it is independent of the above method for reducing the number of cycle values considered.

Shape Descriptors

The correspondence algorithm is independent of how the k boundary points are selected in the first place. To keep our approach as generic as possible, very simple point selection techniques are used. We either choose points at roughly equal intervals along the boundary or points at equal increments of the radial angle (the centroid radii model [Chang *et al.*, 1991]). We refer to the former method as the *perimeter* shape descriptor and the latter as the *radial* descriptor. In some cases, a line from the origin can intercept the boundary at more than one point. To make the radial descriptor consistent, all boundary points close to these intercepts are considered and the point most distant from the origin is selected. This can lead to parts of the shape being ignored. However, the results in Section 3 demonstrate the advantage of using

[1] Discontinuing the likelihood computation for any m whose likelihood drops below a given threshold would further increase the speed of the algorithm.

[2] A random subset of a centered shape will be approximately centered so there is no need to update the center incrementally.

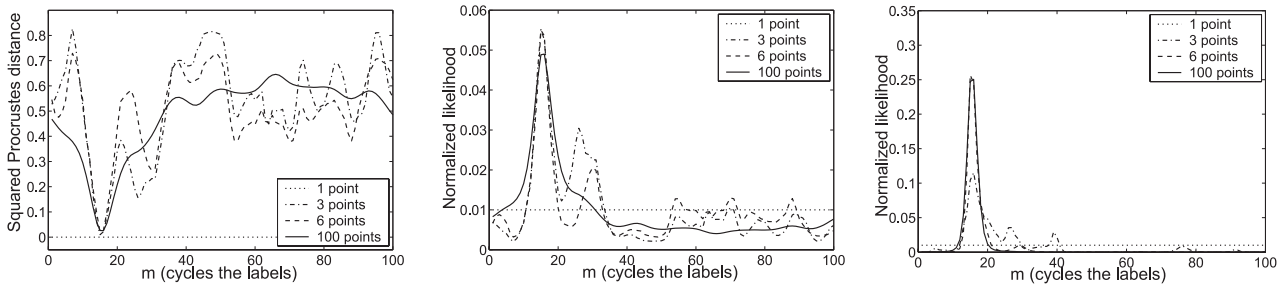

Figure 4: Squared full Procrustes distances (left) and normalized likelihoods (middle) between two shapes from the mouse class. The different lines indicate the distance/likelihood after using a given number of points - or equivalently, computing a given number of columns of $\mathbf{z}\mathbf{w}^*$. Decreasing σ^2 exaggerates the peaked profile of the likelihoods (right).

this sub-optimal descriptor.

2.2 Shape Model

The correspondence technique discussed above can be incorporated into a nearest neighbor classifier: given a test shape, find its optimal correspondence with every training shape and classify on the basis of minimum Procrustes distance. This requires finding a large number of correspondences, a problem that can be avoided using prototypes. Ideally, the *full Procrustes mean* of each class would form the set of prototypes. The definition of this mean follows naturally from the definition of d_F. It is defined as $\bar{\boldsymbol{\mu}} \in \mathbb{C}^k$ such that $||\bar{\boldsymbol{\mu}}|| = 1$ and

$$\bar{\boldsymbol{\mu}} = \arg\min_{\boldsymbol{\mu}} \sum_{l=1}^{n} d_F^2(\mathbf{z}_l, \boldsymbol{\mu}) \tag{4}$$

where $\mathbf{z}_1, \ldots, \mathbf{z}_n$ are all the shapes in the class. Bookstein [Bookstein, 1996] has proposed an iterative method for finding $\bar{\boldsymbol{\mu}}$. By incorporating the correspondence algorithm into this method, we can correspond shapes directly onto a running class mean.[3] This procedure relies on the *full Procrustes fit* for aligning two shapes. Fitting $\mathbf{z}$ to $\mathbf{w}$ transforms $\mathbf{z}$ so that the sum of squared distances between corresponding points is equal to $d_F(\mathbf{z}, \mathbf{w})$. In Fig.1, $d_F(\mathbf{z}, \mathbf{w})$ is zero and a perfect fit is achieved. In Fig.2, $d_F(\mathbf{z}, \mathbf{w})$ is small because the shapes are similar and a reasonable fit is possible. The fit of $\mathbf{z}$ onto $\mathbf{w}$ is given by

$$\mathbf{z}_{fit} = \mathbf{z}^*\mathbf{w}\mathbf{z}/(\mathbf{z}^*\mathbf{z}). \tag{5}$$

The algorithm for finding the correspondences and the mean within a given class is shown in Table 1. Shapes are successively corresponded and fitted to a running mean.

Given a mean, the sample covariance matrices can be calculated at each point. The final *class model* consists of a mean shape - k independent 2D points, each with an associated covariance matrix. Fig.5 shows the algorithm applied to twenty objects from the same class.

2.3 Shape classification

The most likely correspondence and fit between a test shape and each class mean is found as described in Section 2.1. For classification, the model for finding correspondences (eq.(1)) is modified. Points are still assumed independent, but the isotropic covariance is replaced by the sample covariance matrix[4] at each point. Thinking of shapes as sets of points in $\mathbb{R}^2$, a test shape $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_k)^T$ is assigned to a class $c(\mathbf{X})$ using maximum likelihood (ML)

$$c(\mathbf{X}) = \arg\max_{c=1,\ldots,C} (\prod_{j=1}^{k} f(\mathbf{x}_j; \boldsymbol{\mu}_{cj}, \Sigma_{cj})) \tag{6}$$

where C is the number of classes, $(\boldsymbol{\mu}_{c1}, \ldots \boldsymbol{\mu}_{ck})^T \in \mathbb{R}^{k\times 2}$ is the mean of class c, $\Sigma_{cj} \in \mathbb{R}^{2\times 2}$ is the sample covariance at $\boldsymbol{\mu}_{cj}$ and $f(\cdot; \boldsymbol{\mu}_{cj}, \Sigma_{cj})$ is the pdf of the bivariate normal distribution $N(\boldsymbol{\mu}_{cj}, \Sigma_{cj})$.

Table 1: Algorithm for corresponding all shapes from a given class to a running class mean.

Initialize: $\boldsymbol{\mu}$ = final shape in class
for i = 1:cycles for $j = 1$:final shape in class
1. get corresponding labels of shape j and $\boldsymbol{\mu}$
2. fit shape j to $\boldsymbol{\mu}$
3. recompute $\boldsymbol{\mu}$ using arithmetic mean at each point
end
end Fit all shapes to the final $\boldsymbol{\mu}$

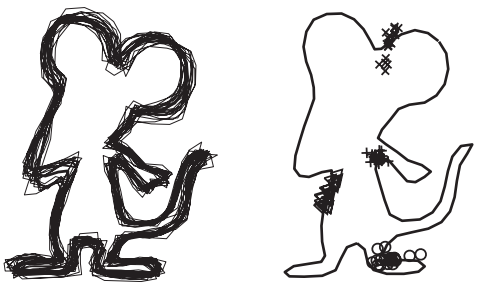

Figure 5: Left: class mean (bold) and the twenty class examples fitted to the mean using 100 points (interpolated for visualization); Right: class mean and scatter of corresponding points at four positions.

[3] $\bar{\boldsymbol{\mu}}$ can be calculated analytically when the correct labels are known(see e.g. [Dryden and Mardia, 1998]).

[4] Computed during the training phase.

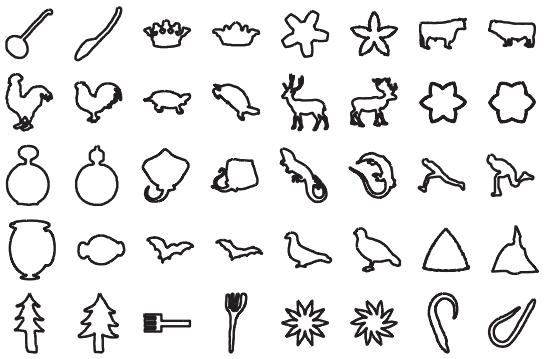

Figure 6: Example boundaries from the MPEG-7 data set.

3 Evaluation and Results

The proposed technique is tested on the benchmark MPEG-7 shape database [Latecki *et al.*, 2000]. This consists of 70 classes with 20 observations in each class. Pixels lying on a closed boundary are extracted in a clockwise direction using Matlab's image processing toolbox. Some of these boundaries are shown in Fig.6 (the same data was used for Figs. 1, 2 and 5). The first principle component of these (2D) points is aligned with the x-axis. This helps reduce the problem of 'sub-interval shifts' producing an artificially high Procrustes distance between similar shapes – particulary for small k. The shape is described by k points using either the perimeter or radial descriptor as described in Section 2.1. Initially we present results without using any speed-up techniques. The impact of these are explored in Section 3.3.

3.1 Leave-one-out Shape Classification

A thirty class subset of the MPEG-7 database was recently used by [Kunttu *et al.*, 2003] in their work on multiscale Fourier descriptors. Their approach outperformed the standard contour Fourier descriptor in various classification tasks. Using leave-one-out classification with a nearest neighbor classifier, they achieved 94.2-96.3% accuracy (depending on the length of shape descriptor). We use the perimeter descriptor with the same data set and testing procedure (i.e. leave-one-out and nearest neighbor) to enable a direct comparison with their results. Test shapes are fitted to training shapes using the correspondence method described in Section 2.1. The nearest neighbor is the training shape closest to the test shape in terms of full Procrustes distance. Results are shown in the first row of Table 2. Classification accuracy is slightly better than that of [Kunttu *et al.*, 2003] and the results are consistent over a wide range of k. The same tests were carried out using the ML method described in Section 2.3. Results are given in the bottom row of Table 2. This prototype based technique is almost as accurate as the nearest neighbor classifier and requires that many fewer correspondences are found - 30 versus 599 per test shape.

The first *column* of Table 3 shows the corresponding results for the full seventy class data set with $k = 48$.[5] One reason for the slightly poorer results on the full data set is that there are now classes for which the perimeter descriptor is definitely not suitable. Despite the generally inferior performance of the radial descriptor (Table 3, middle column), it performs significantly better for some classes. This suggests that using both descriptors may improve classification accuracy. Here, we demonstrate how a naive combination of the descriptors can enhance performance. The ML classifier is modified as follows: Each descriptor (perimeter and radial) gives a vector of class-conditional likelihoods for the test shape. The two vectors are normalized so that they each sum to one. For a given class, the maximum of the two candidate likelihoods is used for classification. This reflects an assumption that the descriptor with the most confident prediction is correct. It is less clear how to normalize the distances for the nearest neighbor classifier. We assume that the majority of shapes bear no resemblance to the test shape and hence, give rise to meaningless correspondences and distances. Normalization is therefore carried over the minimum q entries in each distance vector ($q \ll$ total number of shapes). The distance used for classification is the minimum of the two candidate distances. Here, and in Section 3.2, we fix $q = 200$. The results in Table 3 show a small rise in classification accuracy for the combined case over the perimeter only case. Evidently, the increase in performance is limited by the already high performance of the perimeter descriptor and the simplicity of the normalization technique.

Table 2: Classification performance(%) for the thirty class data set (k = number of points, perimeter descriptor used).

k	12	24	48	72	96
N. neighbor	95.67	97.83	97.83	97.50	98.00
ML	93.17	96.00	96.83	96.67	87.17

Table 3: Classification performance(%) for the seventy class data set with $k = 48$.

	Perimeter	Radial	Combined
N. neighbor	95.71	91.00	96.29
ML	90.36	83.79	92.64

3.2 Bullseye Test for Shape Retrieval

This is a frequently used test in shape retrieval and enables us to compare our algorithm against many of the best performing shape retrieval techniques. A single shape is presented as a query and the top forty matches are retrieved (from the entire data set - the test shape is not removed). The task is repeated for each shape and the number of correct matches (out of a maximum possible 20) are noted. A perfect performance results in $1400 \times 20 = 28000$ matches. Results are given as a percentage of this perfect score. The first two rows of Table 4 show the result of the proposed technique using the perimeter and radial descriptors when no speed-ups were used. The bullseye score was also computed using the combined approach as described for nearest neighbor classification in the previous section. Results are shown in the bottom row of Table 4. [Super, 2004] notes that the best performing algorithms in this test have scored between 75.44% and 78.17%. A naive combination of two simple descriptors

[5] Classification using distance to the nearest prototype is less accurate than the ML classifier. This indicates that the (point-wise) scatter about the mean is sufficiently 'tight' so as to produce covariance matrices that accurately reflect the within-class variability.

Table 4: Scores for the bullseye test(%).

k	12	24	48	72	96
Perimeter	67.29	74.83	76.28	76.00	74.73
Radial	53.33	64.02	67.57	68.03	68.08
Combined	70.67	77.77	79.14	**79.19**	78.56

Table 5: Classification performance using speed-ups.

	Standard	LC	C-F	C-F/LC
NN	97.83	97.13 (8.53)	97.83	97.17 (3.91)
ML	96.83	88.75 (9.73)	94.33	85.00 (4.13)

outperforms all of the previously tested algorithms. It should be noted that these results depend on the free parameter q, the number of shapes used in normalization. We are investigating methods for automatically selecting this value.

3.3 Fast Correspondence

We repeat the classification tests on the thirty class data set using the speed-ups described in Section 2.1. The perimeter descriptor with k=48 is used throughout. The results are given in Table 5 - results are averaged over 10 runs where likelihood cutoff is used. Column one shows the results when no speed-ups are used. The second column shows the results using likelihood cut-off. A correspondence is chosen when the highest likelihood exceeds the second highest likelihood by $2/(2 \times k)$ (i.e. twice the probability assigned to all correspondences under a uniform distribution - recall reflection must be considered). The average number of points needed to choose a correspondence is given in brackets. Column three gives results when the coarse-to-fine correspondence technique is implemented with k'=12. An approximate shape correspondence is mapped back to the 48-point shapes and the best correspondence is chosen over three shifts of the labels in either direction. The final column also shows results using the coarse-to-fine technique but with likelihood cut-off implemented when corresponding the coarse shapes. Implementing the speed-ups has little impact on the accuracy of nearest-neighbor classification. The ML classifier is significantly affected; the coarse-to-fine method produces a moderate reduction in accuracy (94.33 from 96.83) whilst likelihood cut-off leads to a dramatic decrease in performance. Since a test-shape is only compared to its correct class once in the ML case (the class prototype), it is not robust to the increased probability of the correspondence algorithm failing when a low threshold likelihood cut-off is used.

4 Discussion

We have proposed an efficient, correspondence-based shape classification algorithm for 2D boundaries. Shapes are represented by points placed at equal intervals along the boundary; hence, there is no heuristic procedure for choosing the points. A hierarchical matching (going from coarse to fine matches) and likelihood thresholding significantly reduce the cost of finding correspondences with minimal impact on classification accuracy. This ensures that easier to match shapes do not have to go through the complete correspondence computations. The algorithm was tested on the MPEG-7 shape database and was shown to outperform the popular contour Fourier method. Our approach leads to an intuitive class model that can be used for fast prototype-based classification. The results using this method were comparable to that of standard nearest-neighbor classification. The approach was also shown to perform well in the commonly used bullseye retrieval test. Combining two simple shape descriptors further improved retrieval accuracy.

The ideas presented are most relevant to general shape classification and retrieval problems where there is insufficient knowledge to construct a priori models. Combining/choosing between different shape descriptors is difficult in retrieval since there is no training data available. However, relevance feedback (common in document-based information retrieval) is a realistic mechanism for gaining additional information and can be used to guide dynamic selection and weighting of shape descriptors. Future work will investigate this possibility.

References

[Belongie *et al.*, 2002] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Patt. Anal. and Machine Intell.*, 24:509–522, 2002.

[Bookstein, 1996] F.L. Bookstein. Landmark methods for forms without landmarks: morphometrics of group differences in outline shape. *Med. Im. Anal.*, 1(3):225–243, 1996.

[Chang *et al.*, 1991] C.C. Chang, S.M. Hwang, and Buehrer. A shape recognition scheme based on relative distances of feature points from the centroid. *Patt. Recog.*, 24(11):1053–1063, 1991.

[Dryden and Mardia, 1998] I.L. Dryden and K.V. Mardia. *Statistical Shape Analysis*. Wiley, 1998.

[Kent *et al.*, 2004] J.T. Kent, K.V. Mardia, and C.C. Taylor. Matching problems for unlabelled configurations. In *Bioinformatics, Images, and Wavelets*, pages 33–36, 2004.

[Kunttu *et al.*, 2003] I. Kunttu, L. Lepisto, J. Rauhamaa, and A. Viss. Multiscale fourier de scriptor for shape-based image retrieval. In *Proc. of the 12th Int. Conf. on Patt. Recog.*, pages 536–541, 2003.

[Latecki *et al.*, 2000] L. J. Latecki, R. Lakämper, and U. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *IEEE Conf. on Comp. Vis. and Patt. Recog.*, pages 424 –429, 2000.

[Mardia *et al.*, 1979] K.V. Mardia, J.T. Kent, and J.M. Bibby. *Multivariate Analysis*. Academic Press, 1979.

[Super, 2004] B.J. Super. Fast correspondence-based system for shape-retrieval. *Patt. Recog. Lett.*, 25:217–225, 2004.

[Wang *et al.*, 2004] S. Wang, T. Kubota, and T. Richardson. Shape correspondence through landmark sliding. In *IEEE Conf. on Comp. Vis. and Patt. Recog.*, volume 1, pages 143–150, 2004.

[Zhang *et al.*, 2003] J. Zhang, X. Zhang, H. Krim, and G.G. Walter. Object representation and recognition in shape spaces. *Patt. Recog.*, 36:1143–1154, 2003.