In Proceedings of International Conference on Image Processing (ICIP), Atlanta, GA, 2006.

A PROBABILISTIC APPROACH TO ROBUST SHAPE MATCHING

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 probabilistic approach to shape matching that is invariant to rotation, translation and scaling. Shapes can be represented by unlabeled point sets, so discontinuous boundaries and non-boundary points do not pose a problem. Occlusion, significant dissimilarities between shapes and image clutter are explained by a background model, and hence, their impact on the overall match is limited. The ability to operate on incomplete shape representations and ignore part of the input means that, unlike many matching algorithms, our technique performs well on real images. We derive a continuous version of the model which can be used when the 'query shape' is more accurately described by a set of line segments – *e.g.* a boundary polygon or line drawing. The effectiveness of the algorithms is demonstrated using the benchmark MPEG-7 data set and real images.

Index Terms— Shape, object detection, pattern matching

1. INTRODUCTION

Shape-based object recognition is a key problem in computer vision and content-based image retrieval. Over the last decade, numerous shape matching algorithms have been proposed that perform well on benchmark shape retrieval tests. However, many of these techniques match contiguous shape boundaries (*i.e.* the ordering of the boundary points matters) and assume that there are no occlusions, absent features, non-boundary points or clutter. Shapes extracted from real images frequently violate these assumptions, and hence, there is a need for matching algorithms that can operate effectively on partial, noisy shape data.

Techniques that match unlabeled point sets (*e.g.* [1, 2, 3]) are appealing because they make no assumptions as to the nature of the points or the relationships between them (*e.g.* "points 1 and 2 are neighboring boundary points"). The techniques in [4], [5] and [6] use a *soft-correspondence* approach and deal with unmatched points explicitly. This paper makes two contributions in the area of point matching. Firstly, we formulate a probabilistic method based on simple linear transformations that is shown to perform well on real images and a benchmark shape retrieval test. In relation to previous approaches, this most closely resembles the algorithm in [4], but here we use a background model to explain outliers rather than removing them using 'structural editing'. Secondly, we extend the model to the case where one of the shapes is described by a set of line segments. This removes the bias associated with point set representations of lines, and leads to faster matching for shapes composed of relatively few line segments. Just as probabilistic point matching methods can be seen as extensions of the iterative closest point algorithm (ICP) [1] for matching two points sets, our line-based technique extends the ICP for matching a point set to a line set. The arguments justifying a probabilistic approach are the same in both cases: a soft correspondence leads to fewer problems with local minima, outliers are easily handled using a background model, and the variance parameter can be used to indicate the required match quality (or can be annealed during matching).

2. PROBABILISTIC POINT MATCHING (PPM)

For now, let us assume that a shape is represented by a point set of arbitrary size. The points need not belong to the shape boundary and the ordering of the points is irrelevant. Given two such shapes, $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M)^T \in \mathbb{R}^{M\times 2}$ and $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)^T \in \mathbb{R}^{N\times 2}$ (generally $M \neq N$), our task is to compute the correspondence and match between $\mathbf{X}$ and $\mathbf{Y}$. We assume that the $\mathbf{y}_n$ are observations from a mixture model:

$$p(\mathbf{y}_n) = p(\mathbf{y}_n|v_f)p(v_f) + p(\mathbf{y}_n|v_b)p(v_b) \qquad (1)$$

where

$$\begin{aligned} p(\mathbf{y}_n|v_f) &= \sum_{m=1}^{M} p(\mathbf{y}_n|m, v_f)p(m), && (2)\\ \mathbf{y}_n|v_b &\sim Uniform. && (3) \end{aligned}$$

The mixture component v_b represents the 'background model' which ensures that all data points are explained to some extent, and hence, robustifies the model against outliers. The 'foreground' component v_f is the interesting part. The distribution $p(\mathbf{y}_n|v_f)$ is itself a mixture model (eq.(2)), where the $p(\mathbf{y}_n|m, v_f)$ are assumed to be Gaussian. The center of each Gaussian depends on $\mathbf{x}_m$, but unlike standard Gaussian mixture models (GMMs), the movement of the centers is controlled by a single set of transformation parameters s, Γ and $\mathbf{c}$:

$$\mathbf{y}_n | m, v_f \sim \mathcal{N}(s\Gamma\mathbf{x}_m + \mathbf{c}, \sigma^2\mathbf{I}), \tag{4}$$

where s is a scale parameter, $\mathbf{c} \in \mathbb{R}^2$ is a translation vector and Γ is a 2D rotation matrix. We assign equal probability to each mixture component ($p(m)$=$1/M$) rather than learning $p(m)$ from the data. This prevents any single component explaining a large number of $\mathbf{y}_n$ – intuitively, a point of one shape should not explain a large region of the second shape. However, it also prevents $p(m)$ from decreasing for those m that do not explain any data. This seems to have little impact on matching, presumably because the likelihood is maximized by selecting the correct match rather than forcing all components to explain some data.

Note that we could use an additional mixture component m=M+1 for the background model and avoid using the variable v altogether (as used in [7] to match the active sites of proteins). We chose the model specified by eqs.(1-4) because it generalizes easily to the case where the query shape is continuous (Sec. 3). Also, here the background model competes against the mixture model rather than its individual components, allowing a non-outlier to be explained by more than one component.[1] This is desirable since generally there will be no exact 1-1 correspondence and multiple mixture components will contribute towards an individual $\mathbf{y}_n$ – *e.g.* when the true counterpart of a boundary point $\mathbf{y}_n$ lies between two boundary points $\mathbf{x}_m$ and $\mathbf{x}_{m'}$ (see Fig. 1a(ii)).

To recover matches from poor initial alignments yet still find a tight match where possible, we anneal the variance during matching (as in [8]). Maximum likelihood estimates for the other model parameters ($(s, \Gamma, \mathbf{c})$ and $p(v)$) are found using the expectation maximization (EM) algorithm as follows:

E-step: Compute the *responsibilities* using the current parameter values and eq.(4):

$$p(m, v_f | \mathbf{y}_n) = \frac{p(\mathbf{y}_n | m, v_f) p(m) p(v_f)}{p(\mathbf{y}_n)} \tag{5}$$

M-step: Update the *parameters* using the responsibilities:

$$p(v_f) \quad = \quad \frac{1}{N}\sum_n p(v_f|\mathbf{y}_n), \qquad p(v_b) = 1 - p(v_f) \tag{6}$$

$$(s, \Gamma, \mathbf{c}) \quad = \quad \arg\min_{s,\Gamma,\mathbf{c}} \sum_{m,n} p(m, v_f|\mathbf{y}_n) \|\mathbf{y}_n - s\Gamma\mathbf{x}_m - \mathbf{c}\|^2 \tag{7}$$

where $p(\mathbf{y}_n)$=$p(\mathbf{y}|v_b)p(v_b)$+$\sum_m p(\mathbf{y}|m, v_f)p(m)p(v_f)$, and $p(v_f|\mathbf{y}_n)$=$\sum_m p(m, v_f|\mathbf{y}_n)$. Eq.(7) is a *weighted Procrustes matching problem* between two points sets, each of size $N \times M$ – the importance of matching the pair $(\mathbf{x}_m, \mathbf{y}_n)$ is given by

[1]When taking the expectation associated with the EM algorithm, one still assumes that each $\mathbf{y}_n$ was generated by a single (unknown) m.

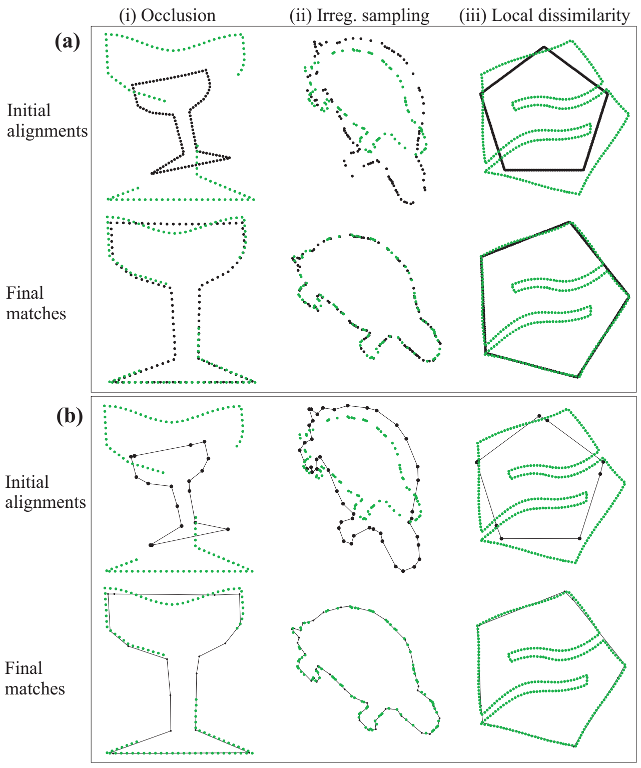

Fig. 1. Examples: (a) PPM, (b) PLM.

$p(m, v_f|\mathbf{y}_n)$. This least squares problem can be solved analytically using a similar approach to that used for unweighted Procrustes problems [4] – *i.e.* the optimal transformation parameters (with respect to the soft correspondence computed during the E-step) can be computed exactly.

Fig. 1a shows how PPM performs on matching problems involving occlusion, irregular sampling and *local dissimilarity* – a feature of one shape is not present or is significantly different on the other shape. The bold points are the $\mathbf{x}_m$ that are collectively transformed to match the faint $\mathbf{y}_n$; the final matches are scaled up for clarity. See that many of the $\mathbf{x}_m$ have no counterparts in (i), whereas it is the $\mathbf{y}_n$ that have no counterparts in (iii). In all cases, the sampling frequency of corresponding sections is different, so there is no perfect point-to-point match.

3. PROBABILISTIC LINE MATCHING (PLM)

In a retrieval scenario, the *query shape/image* may be a sketch, line drawing, or high quality image from which the shape's boundary can be extracted. It is important that the model reflects the continuous nature of the shape in such cases – there should be a continuous tube of probability mass around each line, rather than blobs of probability mass placed at equal intervals along the line (PPM). Also, a model that uses a line-based generating shape is more computationally efficient in cases where few line segments are required.

We focus on the case where a shape's boundary is described by a polygon with k vertices, but the same algorithm can be used whenever one of the shapes is represented by an arbitrary set of line segments. Let $\mathbf{r}(m)$ be the polygon associated with the vertex set $(\mathbf{x}_1, \ldots, \mathbf{x}_K)^T \in \mathbb{R}^{K\times 2}$, where $m \in [0,1)$ is the curve parameter and $0 = m_1 < m_2 < \ldots < m_K < 1$ are the values of m that correspond to the vertices – *e.g.* if the vertex $\mathbf{x}_k$ lies 70% of the way round the perimeter of $\mathbf{r}$, then m_k=0.7. The probabilistic line matching (PLM) model is the same as PPM except that the variable m, which indexes mixture components, is now continuous. Hence, we generate a data point by first selecting a point on the polygon using $p(m)$ and then generating $\mathbf{y}$ from an isotropic Gaussian centered at this point. As before, we assume m is uniformly distributed.

The EM updates for PLM are the same as those for PPM (eqs.(5-7)) except that most of the distributions are now continuous and hence, summations over m become integrals. This creates the following problems. Firstly, we must evaluate

$$p(\mathbf{y}_n|v_f) = \int_0^1 p(\mathbf{y}_n|m, v_f)dm \tag{8}$$

in order to compute $p(\mathbf{y}_n)$. Substituting in the Gaussian density function for $p(\mathbf{y}_n|m, v_f)$ and using $T\mathbf{r}(m) \equiv s\Gamma\mathbf{r}(m) + \mathbf{c}$, we can write

$$\begin{aligned} p(\mathbf{y}_n|v_f) &= \frac{1}{2\pi\sigma^2}\int_0^1 \exp\{\frac{-1}{2\sigma^2}\|\mathbf{y}_n - T\mathbf{r}(m)\|^2\}dm \\ &= \frac{1}{2\pi\sigma^2}\sum_{k=1}^{K}\int_{m_k}^{m_{k+1}} \exp\{\frac{-1}{2\sigma^2}\|\mathbf{y}_n - T\mathbf{r}(m)\|^2\}dm \\ &= \frac{1}{2\pi\sigma^2}\sum_{k=1}^{K}\mathcal{I}_{kn}. \end{aligned}$$

Here, we have split up the integral around the polygon (*i.e.* over m) into segments and defined $\mathcal{I}_{kn}$ to be the integral along the segment between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ (where $\mathbf{x}_{K+1} \equiv \mathbf{x}_1$ and $m_{K+1} \equiv 1$). The expression for $\mathcal{I}_{nk}$ can be rewritten as

$$\mathcal{I}_{kn} = (m_{k+1}-m_k)\int_0^1 \exp\{\frac{-1}{2\sigma^2}\|\mathbf{y}_n - T(\mathbf{x}_k + (\mathbf{x}_{k+1}-\mathbf{x}_k)t)\|^2\}dt,$$

and after some manipulations we find

$$\mathcal{I}_{kn} = \frac{A_{nk}\sqrt{\pi B_k}}{2}[\mathrm{erf}(\frac{1}{\sqrt{B_k}}(1-\lambda_{nk})) - \mathrm{erf}(\frac{-\lambda_{nk}}{\sqrt{B_k}})], \tag{9}$$

where $B_k \equiv 2\sigma^2/(s(m_{k+1}-m_k))^2$, A_{nk} and λ_{nk} are constants that depend on the projection of $\mathbf{y}_n$ onto the line containing $T\mathbf{x}_k$ and $T\mathbf{x}_{k+1}$, and $\mathrm{erf}(\cdot)$ denotes the error function defined as $\mathrm{erf}(z) \equiv \frac{2}{\sqrt{\pi}}\int_0^z e^{-z^2}dz$. Thus, we can evaluate $\mathcal{I}_{kn}$ and hence, evaluate $p(\mathbf{y}_n|v_f)$.

The second difficulty associated with using a continuous mixture model is that we need to solve eq.(7) in the continuous setting:

$$(s,\Gamma,\mathbf{c}) = \arg\min_{s,\Gamma,\mathbf{c}}\sum_n\int_0^1 p(m, v_f|\mathbf{y}_n)\|\mathbf{y}_n - s\Gamma\mathbf{r}(m) - \mathbf{c}\|^2 dm.$$

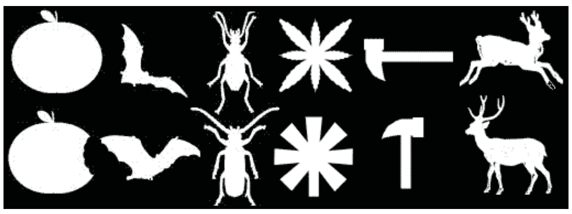
Fig. 2. Example shapes from the MPEG-7 data set.

The method of solution is analogous to the point-based case (the weighted Procrustes problem in eq.(7)) and all integrals can be evaluated using the segment-wise approach described above.

Fig. 1b shows how the PLM model performs on the matching tasks considered earlier. The data shapes are the same, but the generating shapes are polygons. Note that the number of vertices is much smaller than the number of points used in Fig. 1a. In this paper, we use a simple heuristic to select the vertices.

4. SHAPE RETRIEVAL

The "bullseye test" on the MPEG-7 data set[2] has been used extensively to assess the performance of shape retrieval algorithms. The data set consists of 1400 binary images, each containing a single object whose contiguous boundary is easily extracted. There are 70 different classes and 20 observations in each class (Figs. 1 and 2). A shape is presented as a query and the top 40 matches are retrieved from the complete data set. 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 percentage of this maximum score .

We represent each shape's boundary by 100 points. To increase the probability of PPM/PLM finding the correct match (*i.e.* to prevent the EM algorithm from getting trapped in a local minimum), the following procedure is used to initialize the transformation parameters: center $\mathbf{X}$ and $\mathbf{Y}$, normalize for size (where the size of $\mathbf{X}$ is defined as $\sum_m \|\mathbf{x}_m\|^2$), align the principal components, and then consider a range of transformed versions of $\mathbf{Y}$. We identify the transformed $\mathbf{Y}$ that is most similar to $\mathbf{X}$ based on a hard assignment of $\mathbf{y}_n$ to $\mathbf{x}_m$ and the associated transformation parameters are taken as the initial values for PPM/PLM. To apply PLM, the vertices of the polygon $\mathbf{r}(m)$ are chosen from the 100 boundary points using a fast heuristic. The same initial $p(v)$ and σ are used for PPM and PLM, and 16 iterations of EM are used in both cases.[3]

Finally, a similarity score based on PPM/PLM is required. We have found the data likelihood to be an unreliable measure of shape similarity and instead use the following heuristic score function. Each foreground $\mathbf{y}_n$ (*i.e.* $p(v_f|\mathbf{y}_n) > p(v_b|\mathbf{y}_n)$)

[2]http://www.cis.temple.edu/~latecki/research.html#shape

[3]More iterations were required in Figs. 1 and 3 due to the poor initial alignments.

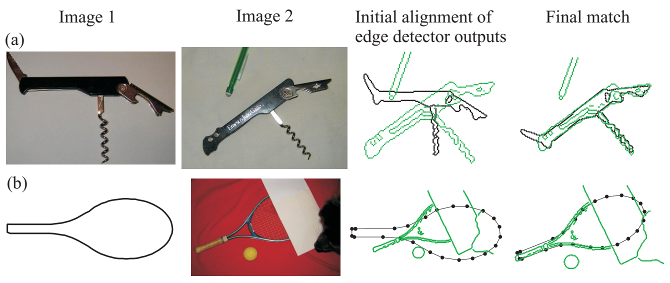

Fig. 3. Matching shapes in real images: (a) PPM, (b) PLM.

is assigned to an $\mathbf{x}_m$ using $p(m|\mathbf{y}_n, v_f)$.[4] The similarity score $\psi_{\mathbf{X}}(\mathbf{Y})$ is simply the number of $\mathbf{x}_m$ that have at least one $\mathbf{y}_n$ assigned to them. This value is dependent on the number of outliers (if j of the $\mathbf{y}_n$ are explained by v_b, the score cannot be greater than 100-j) and how uniformly spread across the $\mathbf{x}_m$ the non-outlying $\mathbf{y}_n$ are. When using PLM, we substitute the polygon for the original 100 point shape after matching and compute $\psi_{\mathbf{X}}(\mathbf{Y})$ as above. The symmetric score function $\Psi(\mathbf{X}, \mathbf{Y}) \equiv \psi_{\mathbf{X}}(\mathbf{Y}) + \psi_{\mathbf{Y}}(\mathbf{X})$ is used for retrieval.

PPM scored 81.98% on the bullseye test, and PLM scored 81.88%. This compares favorably with other algorithms that match unordered point sets: [2]-76.51%, [3]-78.38%, and also with many algorithms that *do* require ordered boundary information – see *e.g.* Table 3 in [9]. Some recently proposed algorithms for matching *ordered* point sets do achieve higher scores. The method in [10] scored 86.35% and lists various other algorithms that have performed very well on this data set. Note that PLM does require ordered boundary information (or at least a set of line segments) for *one* of the shapes. Also, in these experiments, boundary information is used to construct the shape representations since we take points that are equally spaced along the boundary. The key observation is that PPM (and PLM to some extent) remain applicable and effective when no such information is available (Figs. 1(ii) and 3).

For 100 point shapes, it takes $\sim$0.13s to find the the initial alignment and $\sim$0.22s to match shapes using PPM. Selecting the vertices for the polygonal representation takes $\sim$0.01s, and PLM takes between 0.07s and 0.83s depending on the number of vertices used – an average of 35 for this data set. Times are for a 1.4GHz Pentium-M machine using Matlab code. Since we do not check for convergence of the EM algorithm and often use more vertices than necessary for PLM (due to the simplicity of the heuristic for choosing them), it is likely that these times could be significantly reduced.

5. MATCHING SHAPES IN REAL IMAGES

Shapes extracted from real images are often of much poorer quality than the MPEG-7 shapes. It is therefore important that a shape matching algorithm can operate effectively on partial, noisy shape information. In Fig. 3a, the shapes from two real images are matched using PPM. The points with no counterparts (notably the blade in the first image and the pencil and writing on the corkscrew in the second) do not prevent the correct match being found. Fig. 3b shows a 'query by example' problem where PLM matches a template shape to the correct object despite the presence of occlusion and clutter.

[4]This will just be the closest $\mathbf{x}_m$ after matching.

6. SUMMARY AND DISCUSSION

We have presented a probabilistic approach to shape matching which is invariant to rotation, translation and scaling. The algorithm operates on unlabeled point sets of arbitrary size and uses a background model to handle occlusion and significant dissimilarities between shapes. The technique is extended to handle the case when one of the shapes is represented by a set of line segments. Future work will investigate the use of more powerful transformations and extensions of PPM/PLM for learning class models from training data.

7. REFERENCES

[1] P.J. Besl and N.D. McKay, "A method for the registration of 3-d shapes," *PAMI*, vol. 14, no. 2, pp. 239–256, 1992.

[2] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *PAMI*, vol. 24, pp. 509–522, 2002.

[3] C. Grigorescu and N. Petkov, "Distance sets for shape filters and shape recognition," *IEEE Trans. Image. Proc.*, vol. 12, no. 10, pp. 1274–1286, 2003.

[4] B. Luo and E.R. Hancock, "Iterative Procrustes alignment with the EM algorithm," *Image and Vision Computing*, vol. 20, pp. 377–396, 2002.

[5] H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Comp. Vis. and Image Understanding*, vol. 89, pp. 114–141, 2003.

[6] Z. Tu and A.L. Yuille, "Shape matching and recognition using generative models and informative features," in *ECCV*, 2004.

[7] J. T. Kent, K. V. Mardia, and C. C. Taylor, "Matching problems for unlabelled configurations," in *LASR*, 2004.

[8] M Revouw, C.K.I Williams, and G.E Hinton, "Using generative models for handwritten digit recognition," *PAMI*, 1996.

[9] E. Attalla and P. Siy, "Robust shape simlilarity retrieval based on contour segmentation polygonal multiresolution and elastic matching," *Patt. Recog.*, vol. 38, pp. 2229–2241, 2005.

[10] G. McNeill and S. Vijayakumar, "Hierarchical Procrustes matching for shape retrival," in *CVPR*, 2006.