Robust Contour Matching via the Order Preserving Assignment Problem

Clayton Scott and Robert Nowak*
Technical Report TREE 0406
Department of Electrical and Computer Engineering
Rice University
Email: cscott@rice.edu, nowak@engr.wisc.edu

May 14, 2004
Revised, July 15, 2005

Abstract

A common approach to determining corresponding points on two shapes is to compute the cost of each possible pairing of points and solve the assignment problem (weighted bipartite matching) for the resulting cost matrix. We consider the problem of solving for point correspondences when the shapes of interest are each defined by a single, closed contour. A modification of the standard assignment problem is proposed whereby the correspondences are required to preserve the ordering of the points induced from the shapes' contours. Enforcement of this constraint leads to significantly improved correspondences. Robustness with respect to outliers and shape irregularity is obtained by required only a fraction of feature points to be matched. Furthermore, the minimum matching size may be specified in advance. We present efficient dynamic programming algorithms to solve the proposed optimization problem. Experiments on the Brown and MPEG-7 shape databases demonstrate the effectiveness of the proposed method relative to the standard assignment problem.

1 Introduction

A common approach to shape matching proceeds as follows: First, extract a set of features points from each object by, for example, running an edge detector over each image and sampling from the edges. Second, determine pairs of corresponding features in the two feature sets. Third, use the correspondence information to find an aligning transformation such as the least-squares transformation from a certain class (e.g., rigid, affine, thin-plate splines).

The second stage, determining point correspondences, has been the subject of much research. An increasingly popular approach is to build a cost matrix that records the dissimilarity between

*C. Scott is with the Department of Statistics, Rice University, 6100 Main St., Houston, TX 77005. R. Nowak is with the Department of Electrical and Computer Engineering, University of Wisconsin, 1415 Engineering Drive, Madison, WI 53706. This work was supported by the National Science Foundation, grant nos. CCR-0310889 and ANI-0099148, the Office of Naval Research, grant no. N00014-00-1-0390, the Army Research Office, grant no. DAAD19-99-1-0290, and the State of Texas ATP, grant no. 003604-0064-2001. The first author was also supported by an NSF VIGRE postdoctoral training grant.

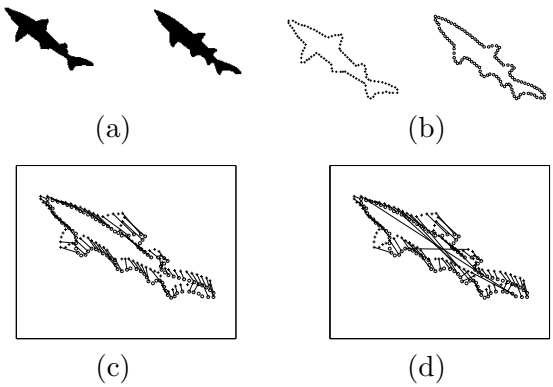


Figure 1: (a) Two shapes in the form of binary images. (b) Representations of those shapes by points sampled from the boundary. (c) An order preserving matching. (d) A non-order preserving matching.

all possible pairs of points on the two shapes. The dissimilarity may be computed, for example, as a distance between shape descriptors associated to the points on each shape. Point correspondences are then given by the solution to the assignment problem (AP) for the constructed cost matrix. Typically, only a fraction of points are assigned counterparts on the other shape, to allow for outliers and irregularities. Recent papers demonstrating the effectiveness of this approach include [1–4].

In this paper we consider the special case where the shapes of interest are defined by a single, closed contour. Our contribution is the formulation and solution of a modification of the standard assignment problem that requires the assignment of corresponding points to preserve the cyclic ordering inherited from the contour. Fig. 1 (c) and (d) show point correspondences that do and do not preserve the ordering of the boundary points, respectively. By enforcing this constraint, the accuracy and geometric fidelity of the point matching is increased, leading in turn to improved registration and shape similarity assessment.

We formulate the cyclic order preserving assignment problem (COPAP) which takes as input an $m \times n$ cost matrix, an outlier tolerance $\epsilon > 0$, and an optional minimum matching size L. We also present two dynamic programming algorithms to solve COPAP. The first algorithm applies when the matching size is unconstrained ($L = 0$), and has an $O(mn \log m)$ implementation. The second algorithm applies when the matching size is constrained ($L > 0$), and has a worst case running time of $O(m^2 nL)$. Experiments with the Brown and MPEG-7 CE-Shape-1 databases demonstrate the improved performance of COPAP versus the standard assignment problem. In our experiments we use the *shape context* as our shape descriptor [1]. We find that COPAP leads to moderate improvements for database retrieval, significant improvements for image registration, and, when $L = 0$, a faster algorithm when compared to the standard assignment problem (which has cubic complexity).

We emphasize the we are not proposing a holistic approach to shape matching, but rather a component of the shape matching process. The input to our algorithm is a cost matrix and the output is a matching. Thus our method can be combined with various edge detectors, shape descriptors, aligning transformations, and so on, provided the shapes are defined by a single contour. MATLAB code (with compiled C-MEX subroutines for COPAP and AP) for reproducing our experiments is available at http://www.dsp.rice.edu/.

1.1 Related Work

The shape matching problem has been considered in various contexts, and a wide range of methodologies have been applied to it. The order-preserving assignment problem developed in this paper is most closely related to three areas of past work.

The first area follows our general setup, but does not assume the feature points are ordered in any way; see [1, 3] and the references therein. These approaches are based on assigning to each feature point a local shape descriptor that captures the spatial relationships with other feature points. The method of [3] uses *labeled distance sets* to capture information about the spatial interrelations of features. The notion of a *shape context*, based on log-polar histograms of feature points, is put forward in [1] and extended in [5] as a robust method for capturing spatial relationships. Labeled distance sets and shape contexts lead naturally to a formulation of the feature correspondence task as an assignment (or bipartite graph matching) problem. The cyclic order-preserving assignment problem that we propose in this paper can be interpreted as a constrained version of the assignment problem.

The second area concerns what can be described as "contour matching" or "curve matching" methods [6–14]. These methods are based on extracting features from contours and then matching the features to gauge the degree of similarity between the two shapes. The referenced approaches differ by the features used to characterize each curve and by the matching algorithm used to determine point correspondences. Typically a dynamic programming algorithm is used to preserve the ordering of features along the contour. Where our work departs from these is in our assumption that correspondences are derived from a cost matrix and/or in our formulation and solution of the point correspondence (assignment) problem.

A third related area includes "structural graph matching" methods, e.g., [15, 16] and the references therein. A key aspect of these methods is that an effort is made to preserve the spatial interrelationships of feature points in the matching process. This can be accomplished using graphical models in conjunction with expectation-maximization or belief propagation algorithms. Our framework may be thought of as a linear graphical model (along the contour) and spatial structure (cyclic ordering) is enforced. Thus, the problem and algorithm in this paper is a special case of structural graph matching which happens to admit an optimal solution in polynomial-time. The general cases [15, 16] require iterative algorithms and convergence to an optimal solution is not guaranteed.

2 Learning point correspondences from a cost matrix

In this section we introduce notation and formulate the problem. Let $x_1, \ldots, x_m$ and $y_1, \ldots, y_n$ denote points on the first and second object, respectively. Assume without loss of generality that $m \leq n$. A cost matrix $C = (c_{ij})$ is an $m \times n$ matrix of non-negative entries, where c_{ij} is a measure of dissimilarity between x_i and y_j. One method of specifying cost matrices is via *shape descriptors*. A shape descriptor may be thought of as a mapping h from the image domain to a high-dimensional vector that encodes shape information of points in the feature set. The cost matrix $C = (c_{ij})$ is then defined by

$$c_{ij} = d(h(x_i), h(y_j)),$$

where d is a function that measures the distance between shape descriptors.

The cost matrix is used to determine a *matching* of feature points. To obtain robustness with respect to outliers, it is common practice to allow some points to not be matched, or, equivalently, to be matched to nonexistent "dummy points." Thus, for our purposes, a matching is a function $\pi : \{1, \ldots, m\} \to \{0, 1, \ldots, n\}$ such that no value in $\{1, \ldots, n\}$ is assumed more than once. Here $\pi(i) = 0$ indicates that the point x_i is not paired with a point on the second shape.

Let $0 \leq L \leq m$ and let Π_L denote the set of all matchings of size at least L, i.e., having at least L non-dummy pairings. The optimal (non-order preserving) matching between the two point sets is defined to be the solution of

$$\min_{\pi \in \Pi_L} \sum_{i=1}^{m} c_{i,\pi(i)}, \tag{1}$$

where $c_{i,0} = \epsilon$ for all i, and ϵ is a user-specified parameter. More is said on choosing L and ϵ below in Section 3.2.

When $L = m = n$, this problem is known as the assignment problem, or the weighted bipartite graph matching problem. It is well studied, and several algorithms exist that produce exact solutions, most (including the fastest) having cubic running time [17]. When $L < m$ or $m < n$, one extends the $m \times n$ cost matrix to an $N \times N$ matrix by padding with ϵ, where $N = m + n - L$. This effectively adds dummy points to each point set. The extended cost matrix is then input to the assignment problem, and the resulting assignment, when restricted to correspondences between non-dummy points, provides a solution to (1).

3 The Cyclic Order Preserving Assignment Problem

Our focus in this paper is a special case of the scenario just described. In particular, we assume that the points in each point set are arranged along a single, closed contour. Further assume that each point set is labeled, starting at 1, in a clockwise manner. We could also label the points counterclockwise, as long as both point sets have the same orientation.[1] The location of the first point does not matter.

With this a priori information about the problem, it makes sense to constrain the matching to preserve the ordering of the contour. Let Π_L^{cyc} denote the set of all matching in Π_L that preserve the cyclic ordering induced by the contours. Thus, if $\pi \in \Pi_L^{cyc}$ and $i_1 < \cdots < i_\ell$ comprise the points not mapped to 0 by π, then there exists k such that $\pi(i_{k+1}) < \cdots < \pi(i_\ell) < \pi(i_1) < \cdots < \pi(i_k)$. We define the cyclic order preserving assignment problem (COPAP) as

$$\min_{\pi \in \Pi_L^{cyc}} \sum_{i=1}^{m} c_{i,\pi(i)}, \tag{2}$$

where again $c_{i,0} = \epsilon$ for all i, as in (1).

The search space for COPAP is exponentially large. For each $L \leq \ell \leq m$, there are $\binom{m}{\ell}$ subsets of the first point set of size ℓ, $\binom{n}{\ell}$ subsets of the second point set of size ℓ, and ℓ cyclic matchings between each such pair of point sets, for a total of $|\Pi_L^{cyc}| = \sum_{\ell=L}^{m} \binom{m}{\ell}\binom{n}{\ell}\ell$ possible matchings to consider. For a typical problem, say $m = n = 100$ and $L = 80$, this number is on the order of 10^{43}. Thus a brute force exhaustive search in infeasible. Yet the search space for COPAP is smaller than

[1] For shape matching in the case where shapes are mirror images of each other, in which case the contours need opposite orientations, we match the first image to the second image and its mirror image, and take the matching with smaller cost.

for AP, which has an $O(n^3)$ solution. This does not imply, however, that it can be solved more quickly. Indeed, because of the special structure of AP, it can be formulated as a special linear program and solved efficiently via primal-dual or other algorithms [18]. We are unaware of any such reduction for COPAP. As we see below, a faster (than cubic) algorithm exists for COPAP when $L = 0$.

3.1 The linear order preserving assignment problem

The basic strategy for solving COPAP involves repeated application of the linear order preserving assignment problem (LOPAP), defined as follows. Take Π_L^{lin} to be the collection of all $\pi \in \Pi_L$ that preserve the linear order of the original point sets. In other words, if $\pi \in \Pi_L^{lin}$ and $i_1 < \cdots < i_\ell$ comprise the points not mapped to 0 by π, then $\pi(i_1) < \pi(i_2) < \cdots < \pi(i_\ell)$. Then LOPAP is the problem

$$\min_{\pi \in \Pi_L^{lin}} \sum_{i=1}^{m} c_{i,\pi(i)}, \tag{3}$$

where, as before, $c_{i,0} = \epsilon$ for all i. Although we omit a formal argument, it should be clear that for some cyclic "shift" of the feature points on the first contour, COPAP and LOPAP have the same solution. Therefore, to solve COPAP, it suffices to solve LOPAP for all such shifts (of which there are m), and take the solution with smallest cost.

3.2 Choosing ϵ and L

Consider matching with AP. When $L = 0$, ϵ has a clear interpretation: Only match pairs of points whose cost is less than ϵ. Thus, ϵ is a threshold on the cost of allowed pairings. When $L > 0$, ϵ has the same interpretation, except that some pairings whose costs exceed ϵ may be forced into the matching to meet the constraint on matching size. Therefore, when the cost of a correct matching is expected to be in a certain range, it makes sense to set $L = 0$ and let the number of outliers (as controlled by ϵ) determine the matching size. In many applications, however, the cost of correct correspondences may vary over a wide range, depending for example on the geometric similarity of the objects. In this case, it makes sense to enforce a minimum matching size $L > 0$. If the shapes of interest are expected to be highly similar, it makes sense to let L be a large percentage of m. If on the other hand, shapes are expected to differ by occluded parts (for example), smaller L may be in order.

With COPAP, the story changes in two respects. First, the interpretation of ϵ as a threshold for outlier rejection is now only approximate. For example, it can happen that all entries in the cost matrix are less than 1, $\epsilon > 1$, and the solution to COPAP has fewer than m matchings. Although perhaps counter-intuitive at first, this results from cyclic order preservation: It may be advantageous to absorb a few higher-cost dummy points into the matching to allow for much lower cost points elsewhere.

Second, COPAP can be solved much more rapidly when $L = 0$ than when $L > 0$. In fact, for large-scale problems, $L = 0$ is the only realistic option. If the fast algorithm is necessary but a minimum matching constraint is still desirable, larger matchings can be obtained by increasing ϵ. Moreover, an iterative bisection search over ϵ can be used to come very close to any desired matching size. With this approach, however, one does lose the interpretation of ϵ as an outlier tolerance.

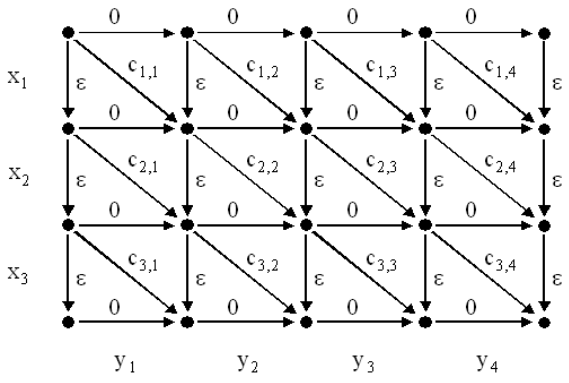


Figure 2: Dynamic programming graph for LOPAP when $L = 0$.

For examples of how ϵ affects database retrieval performance with AP and COPAP, see Fig. 7. Note that costs $c_{i,j}$ derived from shape contexts are between 0 and 1.

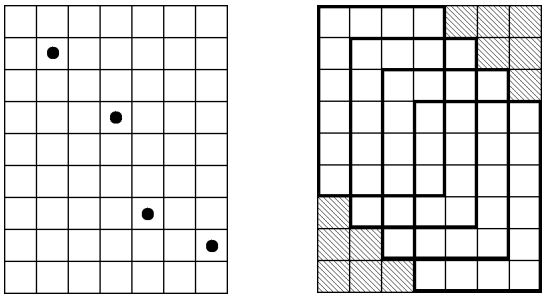
4 COPAP: Unconstrained Matching Size ($L = 0$)

There are two reasons why COPAP is faster when $L = 0$ compared to the case $L > 0$. The first is that LOPAP can be solved in only $O(mn)$ operations compared to $O(mnL)$ for $L > 0$. The algorithm amounts to finding the minimum cost path, from the upper left corner to the lower right corner, through an $(m + 1) \times (n + 1)$ directed graph as illustrated in Fig. 2. In the graph, vertical edges are assigned a cost of ϵ, diagonal edges are assigned entries $c_{i,j}$ from the cost matrix, and horizontal edges are weighted by zero. The optimal path can easily be found via dynamic programming. The algorithm is essentially equivalent, except for the way in which costs are assigned to edges in the graph, to a well known dynamic program for minimizing the "edit distance" between two strings. Note that when $L > 0$, this approach fails because it does not guarantee a minimum matching size.

The second reason why COPAP is faster when $L = 0$ is that, when solving LOPAP at different shifts, previous solutions to LOPAP may be used to constrain the search space. If $1 \leq s_1 < s_2 < s_3 \leq m$ and LOPAP has been solved at shifts s_1 and s_3 of the first point set, then an optimal solution at shift s_2 must lie between the two optimal paths at shifts s_1 and s_3. This allows for a rapid bisection strategy for solving LOPAP at all shifts, and COPAP may be solved in $O(mn \log m)$ time. For a more thorough explanation and justification of this strategy, see [19], who proposed the idea for rapid cyclic matching of strings with respect to edit distance.

5 COPAP: Constrained Matching Size ($L > 0$)

Recall that COPAP reduces to solving LOPAP at m circular shifts of the first point set. For simplicity, we first present the solution to LOPAP in the case $\epsilon = 0$. Observe that under this assumption the optimal matching size must be exactly L. This case is later extended to $\epsilon > 0$.



(a) (b)

Figure 3: Grid representation of LOPAP with $m = 9$, $n = 7$, and $L = 4$. (a) The four dots represent one possible linear order preserving matching. (b) Notice the $L = 4$ bold boxes. The k-th box contains all squares that could possibly be the k-th point in a length L path. Each box is $(m - L + 1) \times (n - L + 1) = 6 \times 4$. The shaded squares could not possibly be part of a length L path.

5.1 LOPAP: The case $\epsilon = 0$

We provide an alternate characterization of LOPAP that aids in visualizing the algorithm. Consider an $m \times n$ grid as depicted in Fig. 3 (a). A black dot at square (i, j) denotes that point i on the first contour is matched to point j on the second contour. We refer to a *path of length* k as a collection of k black dots that progress down and to the right, as in the figure. Paths are denoted by ρ. Let each square on the grid be associated with the cost c_{ij} at the corresponding position in the cost matrix C. The total cost of a path, denoted $t(\rho)$, is the sum of the c_{ij} over points in the path. Observe that paths are in one-to-one correspondence with linear order preserving matchings. Thus, LOPAP (when $\epsilon = 0$) entails finding the path of length L having minimum cost.

The last point in a path, i.e., the point that is furthest down and to the right, is called the terminal point of the path. Let $\rho_{i,j,k}$ denote the path with the smallest cost among all paths of length k that terminate at point (i, j). (Note that $\rho_{i,j,k}$ is undefined if $k < i$ or $k < j$.) Then the solution of LOPAP is $\rho_{i^*,j^*,L}$, where

$$(i^*, j^*) = \arg\min_{i,j} t(\rho_{i,j,L}).$$

Thus, we can solve LOPAP if we can determine $\rho_{i,j,L}$ for each possible i and j. Fortunately, $\rho_{i,j,k}$ may be computed recursively. The base of the recursion is simple: for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, $\rho_{i,j,1}$ is the point (i, j). For $k > 1$, $\rho_{i,j,k}$ is determined as follows: Let (i', j') denote the predecessor to (i, j) in the path $\rho_{i,j,k}$. Observe that

$$(i', j') = \arg\min_{r<i, s<j} t(\rho_{r,s,k-1}).$$

Then $\rho_{i,j,k}$ is simply $\rho_{i',j',k-1}$ with the point (i, j) appended.

In the above formulation, determining (i', j') requires a search over all points above and to the left of (i, j), which in general requires considering $O(mn)$ different options. However, it is possible to limit the scope of the search to 3 options, thus giving rise to a much faster algorithm. The reader not interested in these details may skip the next paragraph.

We know that $\rho_{i,j,k}$ terminates at point (i,j) on the grid, by definition. There are three possibilities for (i',j'): (a) $i' = i-1$ and $j' = j-1$; (b) $i' \leq i-1$ and $j' < j-1$; and (c) $i' < i-1$ and $j' \leq j-1$. It is not important that cases (b) and (c) are not mutually exclusive. The value of $t(\rho_{i',j',k-1})$ is determined by these three cases as follows: (a) $t(\rho_{i',j',k-1}) = t(\rho_{i-1,j-1,k-1})$; (b) $t(\rho_{i',j',k-1}) = t(\rho_{i,j-1,k}) - c_{i(j-1)}$; and (c) $t(\rho_{i',j',k-1}) = t(\rho_{i-1,j,k}) - c_{(i-1)j}$. We can determine which case holds by finding the smallest of these three expressions (which are based on previously computed quantities). If (a) is smallest, then $(i',j') = (i-1,j-1)$. If (b) is smallest, then (i',j') is the predecessor of $(i,j-1)$ in $\rho_{i,j-1,k}$. If (c) is smallest, then (i',j') is the predecessor of $(i-1,j)$ in $\rho_{i-1,j,k}$. If both (b) and (c) are minimal, the same value of (i',j') will usually be the same, and even if not, the cost of the final path will be unchanged.

5.2 Computational Complexity

The complexity of the above algorithm is dominated by the computation of $\rho_{i,j,k}$ for $1 \leq i \leq m, 1 \leq j \leq n$, and $1 \leq k \leq L$. At first glance, there appear to be mnL such quantities to compute. However, $\rho_{i,j,k}$ is defined only for certain combinations of i, j, and k. For example, the last point in any path of length L can be no fewer than $L-1$ rows from the top of the grid, and no fewer than $L-1$ columns from the left side of the grid. Thus, we need only compute $\rho_{i,j,L}$ for $L \leq i \leq m$ and $L \leq j \leq n$. In general, we need to compute $\rho_{i,j,k}$ for $k \leq i \leq m-L+k$ and $k \leq j \leq n-L+k$. These regions are represented by the boxes in Fig. 3 (b).

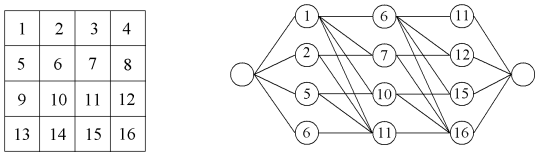
This gives a reduction from mnL to $(m-L+1)(n-L+1)L$ in the number of updates of $\rho_{i,j,k}$. Each such update takes a fixed amount of processing, so the overall runtime of LOPAP is $O((m-L+1)(n-L+1)L)$. Thus, when L is very close to m and n, the running time can be as low as $O(m)$. If L is a fixed fraction of m and n, the reduction does not affect the order of the computational complexity, although it does significantly decrease the leading constant.

In practice, with $m = n = 100$ and $L = 85$, COPAP is solved in about 0.45 seconds on a 3 GHz processor. This compares with about 10 ms to solve the standard assignment problem using the $O(n^3)$ algorithm of Jonker and Volgenant [20]. The large discrepancy is due in large part to the $O(mnL)$ memory requirements of COPAP. When $L = 0$, the faster algorithm of Section 4 executes in 10 ms, about the same as AP. This algorithm is faster than AP for larger m and n.

5.3 Trellis Formulation

The algorithm for LOPAP ($L > 0$) may be viewed as the Viterbi algorithm on a certain trellis. Fig. 4 shows a simple example. Each node in the trellis corresponds to a square on the grid, and hence to a particular pair of points on the two shapes. In particular, the nodes in the k-th column of the trellis correspond precisely with those squares (i,j) in the grid representation such that $k \leq i \leq m-L+k$ and $k \leq j \leq n-L+k$. These are the same points described by the bold boxes in Fig. 3 (b), although the dimensions of the problem are different there.

Suppose (i',j') and (i,j) are pairings corresponding to nodes in columns $k-1$ and k of the trellis. Then there is a link between those two nodes if and only if $i' < i$ and $j' < j$. A path through the trellis thus corresponds to a path on the grid, and to a linear order preserving assignment. With this setup, LOPAP may be viewed as finding the shortest (or least expensive) path through the trellis, where the cost of the path is the cost of the corresponding assignment. The Viterbi algorithm is a well known dynamic programming procedure for determining shortest paths through



(a) (b)

Figure 4: Trellis representation of LOPAP with $m = n = 4$ and $L = 3$: (a) Enumeration of grid. (b) Trellis showing all possible length 3 paths. LOPAP seeks the path through this trellis with smallest cost.

trellises [21]. A brief inspection reveals that our algorithm coincides with the Viterbi algorithm on this trellis.

5.4 LOPAP: The case $\epsilon > 0$

It is straightforward to extend the algorithm proposed in Section 5.1 to the general case $\epsilon > 0$. From that algorithm, we know how to construct optimal paths ρ_ℓ^* of size $\ell = L, \ldots, m$. Because of the recursive nature of the algorithm, computations may be shared in computing these optimal paths. They may be constructed in $O(mnL)$ operations, the same as for computing a single optimal path.

Having determined the optimal paths ρ_ℓ^* of size $\ell = L, \ldots, m$, the solution ρ^* to LOPAP with $\epsilon > 0$ is given by

$$\rho^* = \underset{\rho_\ell^* : \ell = L, \ldots, m}{\arg\min} \; t(\rho_\ell^*) + (m - \ell)\epsilon.$$

Recall that $t(\rho)$ denotes the cost of a path/matching, i.e., the sum of the costs of the individual pairings in the path. In sum, the general algorithm for COPAP is $O(m^2 nL)$, or more precisely, in light of the discussion in Section 5.2, $O(m(m - L + 1)(n - L + 1)L)$.

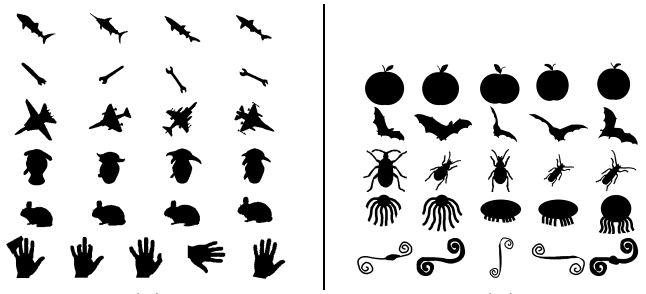
6 Experiments

We conduct experiments on two shape databases. In each experiment, we compare COPAP with the standard assignment problem (AP) in two regards: database retrieval performance and usefulness in designing aligning transformations. Contours were extracted using the MATLAB function `contourc`. If multiple contours exist for a given shape, the largest is taken and remaining points are discarded. For a shape descriptor we employ the rotationally-invariant shape context[2] [1]. The cost matrix C is formed by computing the chi-squared distance between pairs of shape contexts on the two contours, as in [1]. The (dis)similarity between shapes is defined to be the overall cost of the optimal assignment.

6.1 The Brown shape database

We present two experiments based on the collection of 25 shapes shown in Fig. 5 (a) and obtained from the Brown computer vision group web site [22]. Each row of the figure is considered to be a

[2]Here rotationally-invariant means that the log-polar histogram is oriented with respect to the tangent of the contour.



(a) (b)

Figure 5: (a) All shapes from the Brown database. (b) Representative shapes from the MPEG-7 database.

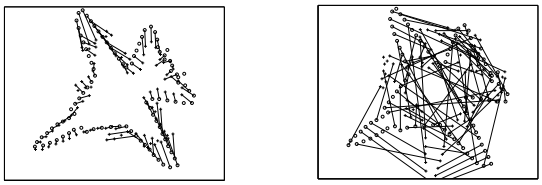
separate class. In both experiments, each shape in the database is matched to every other shape, and a measure of similarity is assigned. For each shape, the top three most similar shapes are identified. Performance is evaluated by counting the number of first, second, and third ranked matches that belong to the same class as the test pattern.

We uniformly sample 100 points from the contours of each shape, enforce matchings of size $L = 85$, and set the outlier parameter at $\epsilon = 0$. This experiment was performed in [1] to demonstrate rotationally invariant shape matching with shape contexts. The number of correct first, second, and third ranked matches achieved was 25/25, 24/25, and 22/25 (we have verified this result). We repeat this experiment, using COPAP instead to compute the optimal assignment. Our results are 25/25, 24/25, 23/25. Results of previous studies are 23/25, 21/25, 20/24 [22]; 25/25, 21/25, 19/25 [8]; and 25/25, 24/25, 25/25 [5]. The improved performance of [5] with respect to our method can be attributed entirely to the shape descriptor; see the next section for further discussion.

The above experiment indicates that, when using the assignment cost for database retrieval, COPAP offers a modest improvement over the standard assignment problem. When we take a closer look at the matchings found by the two methods, however, the benefits of preserving the cyclic ordering are considerably greater.

A second experiment repeats the one above, but uses a different measure of similarity that better captures the geometric quality of the correspondences. In particular, we solve for the optimal assignment between two shapes (as before) using AP and COPAP, and estimate a geometric transformation based on the matching. Instead of using the matching cost to quantify similarity, we now use the average squared error between transformed points on the first shape and their corresponding points on the second. The transformation is the least squares rigid body transformation (variable translation and rotation). Using COPAP to determine point correspondences, the retrieval numbers are 25/25, 25/25, 21/25. Using AP, the numbers are 21/25, 19/25, 14/25. Nearly all of the errors when using AP arise from classes that require rotational invariance (e.g., tools and airplanes). See for example the matching produced by COPAP and AP in Fig. 6.

We offer two possible explanations for this gap in performance. First, rotation invariant shape matching with shape contexts requires the estimation of tangents to the contours, which can be difficult for contours with corners. Second, several of the shapes (e.g., tools and airplanes) possess symmetry, or "near-symmetry." In other words, some parts of a shape (e.g., the nose of the airplane) look like other parts (e.g., the wing), as far as the shape descriptor is concerned. This leads to the behavior in Fig. 6 (b), were AP assigns points on the nose of one airplane to points on the nose *and* wing of the other. In contrast, such a matching is impossible with COPAP. Neighboring points



(a) (b)

Figure 6: Aligning two airplanes: (a) Matching and registration produced by COPAP. (b) Matching and registration produced by the standard assignment problem. Unlike COPAP, neighboring points are not required to match neighboring points.

ϵ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
COPAP	76.56	80.15	80.85	81.20	81.43	81.81	82.00	82.22	82.34	82.46
AP	76.51	79.39	79.14	78.53	78.22	77.96	77.81	77.72	77.71	77.70

Figure 7: Retrieval rates (in %) of COPAP and AP for the bullseye test on the MPEG-7 shape database.

are forced to match neighboring points, thus helping to overcome problems arising from estimating tangents (in the case of shape contexts) and "near-symmetries."

As a final note, we remark that all of COPAP's third rank errors involved the pattern of the F-14 Tomcat (the rightmost plane in Fig. 5 (a)) . As illustrated in Fig. 6 (a), this is not due to a poor registration, but simply to the width of the aircraft. With a more flexible transformation model, we might expect to avoid this problem.

6.2 The MPEG-7 shape database

Our second set of experiments look at the MPEG-7 shape database for Core Experiment CE-Shape-1 part B [23], depicted in Fig. 5 (b). The database consists of 1400 shapes, 20 shapes in 70 categories. Performance is measured by the so-called "bullseye test": Each shape is compared to every other shape in the database, and the number of correct matches in the top 40 is counted. There are 20 possible correct matches per shape queried. The retrieval rate is the total number of correct matches divided by the total number possible, $20 \times 1400 = 28000$. Previous studies have obtained rates of 75.44% [9], 76.45% [23], 76.51% [1], 77.76% [24], 76.93% [25], 78.17% [11], 79.19% [4], 80.03% [26], 78.38% [3], 58.50% [13], and 85.40% [5].

We uniformly sample 100 points from the contours of each shape and take $L = 0$ to allow the use of the fast algorithm of Section 4. Since some of the shapes in the database exhibit mirror symmetry, when comparing two shapes, we match to the original image and its mirror image, and use the matching with the smaller cost. Retrieval rates of COPAP and AP for various ϵ are shown in Fig. 7. Fig. 8 shows recognition rate (fraction of correct matches) as a function of similarity rank for $\epsilon = 0.4$.

A clear trend emerges: As ϵ increases, COPAP improves while AP performs worse. Recall that as ϵ increases, larger matchings result. These results are explained by the fact, discussed in Section 6.1 and again below, that COPAP is more likely than AP to produce a geometrically faithful matching. If a matching is geometrically faithful, increasing its size will increase the discriminatory impact of the matching, whereas the opposite effect results for a geometrically unfaithful matching.

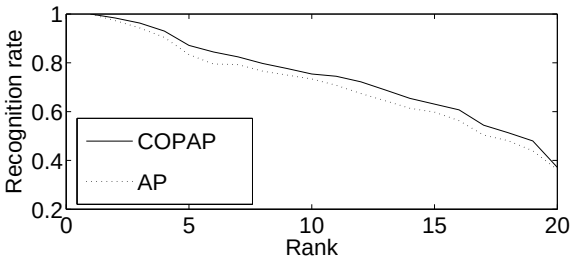


Figure 8: Average recognition rate as a function of similarity rank, when $\epsilon = 0.4$.

Only one previous method outperforms our approach based on rotationally invariant shape contexts together with COPAP. Moreover, the better performance of [5] may be attributed to the shape descriptor. To achieve their reported rate, they employ a modified shape context based on "inner distances." Cyclic matchings are enforced, but by means of a suboptimal variant of the algorithm described in Section 4, wherein the optimal linear assignment is computed at 4 equally spaced shifts. It seems reasonable to expect that combining our algorithm for exact cyclic matching with their shape descriptor would yield even better results.

It is interesting to note that shape contexts with 100 points and unconstrained matching outperforms the method of [1], where shape contexts based on 300 points per shape are used in conjunction with metrics based on appearance similarity and deformation energy. Part of the reason may be that we use rotationally invariant shape contexts, whereas [1] does not employ the rotationally invariant option on this data set. But perhaps a greater reason for the discrepancy is that AP does not always yield geometrically plausible correspondences, This leads to poor registration, and hence misleading costs due to appearance and deformation energy. To test this hypothesis, we took several pairs of shapes from the same class and computed the least squares linear conformal transformation (variable translation, rotation, and scale) based on the optimal assignment. The results (not shown) consistently resembled those of Fig. 6, wherein COPAP produces an accurate registration while AP produces nonsense.

7 Summary and Conclusions

We introduce and solve the cyclic order preserving assignment problem (COPAP), which can be used to find a cyclic order preserving matching between two point sets taken from shapes defined by a single close contour. COPAP has two free parameters, the minimum matching size L and the cost ϵ assigned to outliers. The general algorithm has $O(m^2nL)$ complexity and involves solving a linear order preserving assignment problem (LOPAP) at different shifts. The algorithm for LOPAP amounts to computing the shortest path through a certain trellis, and as such can be seen as an instance of the Viterbi algorithm. In the case $L = 0$, a much faster $O(mn \log m)$ algorithm for COPAP exists, facilitating the use of cyclic matching for large-scale problems.

Our experiments reveal the effectiveness of cyclic matchings. Using the shape context as our shape descriptor, we demonstrate improved performance versus non-order preserving matchings. The improvement is moderate when using the cost of the optimal assignment for database retrieval, but major when performing image registration or using geometric transformations for database retrieval. Indeed, the standard assignment problem can result in completely meaningless registrations,

whereas COPAP seems much less susceptible to this occurrence.

References

[1] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, 2002.

[2] H. Chui, "Non-rigid point matching: Algorithms, extensions and applications," Ph.D. dissertation, Yale University, May 2001.

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

[4] G. McNeill and S. Vijayakumar, "2D shape classification and retrieval," in *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI '05)*, Edinburgh, UK, 2005.

[5] H. Ling and D. Jacobs, "Using the inner-distance for classification of articulated shapes," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, San Diego, CA, 2005.

[6] L. J. Latecki and R. Lakämper, "Shape similarity measure based on correspondence of visual parts," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 22, no. 10, pp. 1185– 1190, 2000.

[7] N. Ueda and S. Suzuki, "Learning visual models from shape contours using multiscale convex/concave structure matching," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 15, no. 4, pp. 337–352, 1993.

[8] Y. Gdalyahu and D. Weinshall, "Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 21, pp. 1312–1328, 1999.

[9] F. Mokhtarian, S. Abbasi, and J. Kittler, "Efficient and robust retrieval by shape content through curvature scale space," in *Image Databases and Multi-media search*, A. W. M. Smeulders and R. Jain, Eds. World Scientific, 1997, pp. 51–58.

[10] D. Tell and S. Carlsson, "Combining appearance and topology for wide baseline matchine," in *Computer Vision - ECCV 2002 : 7th European Conference on Computer Vision*, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Heidelberg: Springer-Verlag, 2002.

[11] T. Sebastian, P. Klein, and B. Kimia, "On aligning curves," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 25, no. 1, pp. 116–124, 2003.

[12] E. Milios and E. Petrakis, "Shape retrieval based on dynamic programming," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 141–147, 2000.

[13] I. Bartolini, P. Ciaccia, and M. Patella, "WARP: Accurate retrieval of shapes using phase of Fourier descriptors and time warping distance," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 27, no. 1, pp. 142–147, 2005.

[14] S. Wang, T. Kubota, and T. Richardson, "Shape correspondence through landmark sliding," in *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2004, pp. 143–150.

[15] B. Luo and E. R. Hancock, "Structural graph matching using the EM algorithm and singular value decomposition," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 23, pp. 1120–1136, 2001.

[16] J. M. Coughlan and S. J. Ferreira, "Finding deformable shapes using loopy belief propagation," in *Proc. European Conference on Computer Vision, LNCS 2352*. Springer-Verlag, 2002, pp. 453–468.

[17] M. Dell'Amico and P. Toth, "Algorithms and codes for dense assignment problems: The state of the art," *Discrete Applied Mathematics*, vol. 100, pp. 17–48, 2000.

[18] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization : Algorithms and Complexity*. Englewood Cliffs, N.J.: Prentice Hall, 1982.

[19] M. Maes, "On a cyclic string-to-string correction problem," *Inform. Process. Let.*, vol. 35, pp. 73–78, 1990.

[20] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, pp. 325–340, 1987.

[21] D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[22] D. Sharvit, J. Chan, H. Tek, and B. Kimia, "Symmetry-based indexing of image databases," *J. Visual Communication and Image Representation*, vol. 9, no. 4, pp. 366–380, 1998.

[23] L. J. Latecki, R. Lakämper, and U. Eckhardt, "Shape descriptors for non-rigid shapes with a single closed contour," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2000, pp. 424–429.

[24] T. Adamek and N. O'Connor, "Efficient contour-based shape representation and matching," in *MIR '03: Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval*. New York, NY, USA: ACM Press, 2003, pp. 138–143.

[25] B. Super, "Fast correspondence-based system for shape retrieval," *Pattern Recognition Letters*, vol. 25, no. 2, pp. 217–225, 2004.

[26] Z. Tu and A. Yuille, "Shape matching and recognition using generative models and informative features," in *Proc. 8th European Conference on Computer Vision*, 2004.