On Aligning Curves

Thomas B. Sebastian, Philip N. Klein, Benjamin B. Kimia

Abstract— **We present a novel approach to finding a correspondence (alignment) between two curves. The correspondence is based on a notion of an *alignment curve* which treats both curves symmetrically. We define a similarity metric based on the alignment curve using two intrinsic properties of the curve, namely, length and curvature. The optimal correspondence is found by an efficient dynamic-programming method both for aligning pairs of curve segments and pairs of closed curves, and is effective in the presence of a variety of transformations of the curve. Finally, the correspondence is shown to serve as a key element in handwritten character recognition, prototype formation and object recognition, and is potentially useful in other applications such as registration and tracking.**

Keywords—**Curve alignment, Recognition, Dynamic Programming, Prototypes.**

I. INTRODUCTION

This paper presents a novel method to find the optimal correspondence or *alignment* between 2D curves using their intrinsic properties. This alignment has served as a key element in many applications such as object recognition based on silhouettes [1], [2], [3], handwritten character recognition [4], [5], [6], tracking [7], [8], *etc.* While existing curve alignment/matching approaches typically take advantage of constraints pertaining to their particular domain of application, we approach this problem generically, keeping in mind that domain-specific constraints can be added subsequently. We first review current approaches for curve alignment and then discuss how we extend current techniques and present an overview of our approach.

Current curve alignment methods can be classified into two categories: methods based on rigid transformations [1], [9], [10], and those based on non-rigid deformations [7], [8], [11], [12]. Methods based on *rigid transformations* rely on matching feature points by finding the optimal rotation, translation, and scaling parameters [1], [10]. Since these methods assume that the outlines can be aligned by a rigid transformation, they are sensitive to articulations, deformations of parts, occlusion, and other variations in the object form. Methods based on non-rigid *deformations* model articulation and other deformations by finding the mapping from one curve to another that minimizes a performance functional consisting of "stretching" and "bending" energies [7], [8], [11], [12]. The minimization problem in the discrete domain is transformed into one of matching shape signatures with curvature, bending angle, or absolute orientation as attributes [5], [2], [3]. These methods suffer from one or more of the following drawbacks: asymmetric treatment of the two curves, sensitivity to sampling of the curves, lack of rotation and scaling invariance, and sensitivity to occlusion and articulations. We explore an extension of these approaches that address some of these issues.

Our approach to finding the optimal correspondence between 2D curves relies on using the *intrinsic* properties of the curves in an energy minimization framework as in [7], [12]. However, a key problem is the *asymmetric treatment* of the curves being matched. Tagare [8] handles this issue by introducing a bimorphism and searching in the space of pairs of functions. Observing a redundancy in this search, we introduce the notion of an *alignment curve* which ensures a symmetric treatment by searching in the space of a single function, Section II. This naturally leads to an "edit distance" metric. Efficient algorithms for finding the optimal alignment using dynamic programming are described in Section III for matching both curve segments and closed curves. Section IV illustrates the generic nature of the proposed curve alignment framework in several applications including handwritten character recognition, prototype formation, and object recognition.

T. Sebastian and B. Kimia are with the Division of Engineering, Brown University, Providence, RI. E-mail: tbs@lems.brown.edu and kimia@lems.brown.edu.

P. Klein is with Department of Computer Science, Brown University, Providence, RI. E-mail: klein@cs.brown.edu.

II. CURVE ALIGNMENT

This section discusses the mathematical formulation of the problem of aligning two curves. We first consider the case of aligning two curve segments and then use this to align two closed curves.

A. Deformation-based Approach

Cohen *et al.* [7] pioneered the deformation-based approach to curve matching. The basic premise of their approach was to match high curvature points along the curves, while maintaining a smooth displacement field. The problem was cast in terms of minimizing an energy functional penalizing "bending" and "stretching", in a physical analogy similar to the one used in formulating active contours [13]. Younes [12] formalized the approach. Tagare *et al.* [8] pointed out the inherent asymmetry in the treatment of the two curves and proposed a "bimorphism" that ensures a symmetric formulation. We now give the specifics of the problem formulation.

Denote the curve segments to be matched by $\mathcal{C}(s) = (x(s), y(s)),\ s \in [0, L]$ and $\bar{\mathcal{C}}(\bar{s}) = (\bar{x}(\bar{s}), \bar{y}(\bar{s})),\ \bar{s} \in [0, \bar{L}]$, where s is arc length, x and y are coordinates of each point, L is length of $\mathcal{C}$, and each is similarly defined for $\bar{\mathcal{C}}$. A central premise of this approach is that the "goodness" of the optimal match is the sum of the "goodness" of the matches between corresponding subsegments. This allows an energy functional to convey the goodness of a match as a function of the correspondence or alignment of the two curves as proposed earlier [7], [12]. Let a mapping $g : [0, L] \to [0, \bar{L}]$, $g(s) = \bar{s}$, represent an alignment of the two curves. Specifically, Cohen *et al.* [7] compare the displacement velocities and bending energies in the form of

$$\mu[g] = \int_{\mathcal{C}} \left|\frac{\partial}{\partial s}(\bar{\mathcal{C}}(\bar{s}) - \mathcal{C}(s))\right|^2 ds + R\int_{\mathcal{C}} (\kappa(s) - \bar{\kappa}(\bar{s}))^2 ds, \tag{1}$$

where κ and $\bar{\kappa}$ are the curvatures along the curves $\mathcal{C}$ and $\bar{\mathcal{C}}$, respectively, R is a parameter, and $\bar{s} = g(s)$. Younes [12] uses a similar functional. Note that these methods are not invariant to the relative *rotation* of the two curves. Hence, the optimal relative orientation must also be found. Also, in Cohen's method, the cost of the optimal deformation is not invariant to the way the curves are sampled, due to the curvature comparison term. We now formulate the problem in an intrinsic manner which addresses both issues.

B. Intrinsic Formulation

Definition 1: Let $\mathcal{C}\big|_{[s_1,s_2]}$ denote the portion of the curve from s_1 to s_2, and let $g\big|_{([s_1,s_2],[\bar{s}_1,\bar{s}_2])}$ denote the restriction of the mapping g to $[s_1,s_2]$, where $\bar{s}_i = g(s_i)$, $i = 1,2$. Define a measure μ on this alignment function, $\mu[g]\big|_{([s_1,s_2],[\bar{s}_1,\bar{s}_2])}$: $g\big|_{([s_1,s_2],[\bar{s}_1,\bar{s}_2])} \to \mathbf{R}^+$, to denote the cost of deforming $\mathcal{C}\big|_{[s_1,s_2]}$ to $\bar{\mathcal{C}}\big|_{[\bar{s}_1,\bar{s}_2]}$.

We restrict this measure μ to one which satisfies an *additivity property*

$$\mu[g]\big|_{([s_1,s_3],[\bar{s}_1,\bar{s}_3])} = \mu[g]\big|_{([s_1,s_2],[\bar{s}_1,\bar{s}_2])} + \mu[g]\big|_{([s_2,s_3],[\bar{s}_2,\bar{s}_3])},$$
$$\forall s_1 \le s_2 \le s_3 \in [0,L], \forall \bar{s}_1 \le \bar{s}_2 \le \bar{s}_3 \in [0,\bar{L}], \tag{2}$$

where $\bar{s}_i = g(s_i)$, $i = 1,2,3$. This property implies that the match process can be decomposed into a number of smaller matches, and can be written as a functional

$$\mu[g]\big|_{([0,L],[0,\bar{L}])} = \int_0^L \mu[g]\big|_{([s,s+ds],[g(s),g(s+ds)])} ds. \tag{3}$$

The optimal match is then given by $g^* = \underset{g}{\operatorname{argmin}}\, \mu[g]\big|_{([0,L],[0,\bar{L}])}$.

Consider two infinitesimal curve segments $\mathcal{C}\big|_{[A,B]}$ and $\bar{\mathcal{C}}\big|_{[\bar{A},\bar{B}]}$ of lengths ds, $d\bar{s}$, and curvatures κ, $\bar{\kappa}$, respectively. Since we compare the intrinsic aspects of these curve segments, we align their start points A and $\bar{A}$ and their respective tangents $\vec{T}_A$ and $\vec{T}_{\bar{A}}$, Figure 1a. The cost of matching the infinitesimal curve segments is the degree by which the endpoints B and $\bar{B}$ differ, which can be formulated as

$$\mu[g]\big|_{([s_1,s_1+ds],[\bar{s}_1,\bar{s}_1+d\bar{s}])} = |d\bar{s} - ds| + R|d\bar{\theta} - d\theta|, \tag{4}$$

where R is a constant[1]. Then, the resulting functional is given by

$$\begin{aligned}\mu[g] = & \int_{\mathcal{C}} \left[\left|\frac{d\bar{s}}{ds} - 1\right| + R\left|\frac{d\bar{\theta}(\bar{s})}{d\bar{s}}\frac{d\bar{s}}{ds} - \frac{d\theta(s)}{ds}\right|\right] ds \\ = & \int_{\mathcal{C}} \left[\left|g'(s) - 1\right| + R\left|\bar{\kappa}(g(s))g'(s) - \kappa(s)\right|\right] ds,\end{aligned} \tag{5}$$

where the first term penalizes "stretching" and the second term penalizes "bending". This is the same as the linear cost model of Basri *et al.* [11][2]. Observe that while this formulation is symmetric in form [11], the explicit dependence on the alignment function g makes it inherently asymmetric [8]. Specifically, algorithms which are based on differentiable mapping of one curve to the other are asymmetric, as the inverse map may not be one-to-one or differentiable. Tagare [8] instead propose a "bimorphism", which diffeomorphically maps the pair of curves to be matched. Note that the bimorphism corresponds to a closed curve in the space of $\mathcal{C}_1 \times \mathcal{C}_2$. Specifically, he formulates a cost function that minimizes differences in local length and orientation changes $|d\bar{s} - ds|$ and $|d\bar{\theta} - d\theta|$ along each differential segment of this curve, and seek a *pair* of functions ϕ_1 and ϕ_2 (elements of the bimorphism) which optimize this cost functional.

C. Symmetric Formulation: Alignment Curve

We approach this asymmetry issue by observing that the formulation based on Equation 5 does not allow the mapping of a single point in the first curve to a segment in the second curve. This is because the notion of an alignment is captured by a (univalued) function g. To alleviate this difficulty we revise the formulation. Reconsider an alignment between two curves as a pairing of two particles, one on each curve traversing their respective paths monotonically, but with finite stops allowed. This alignment can be specified in terms of two functions h and $\bar{h}$ relating arc length along $\mathcal{C}$ and $\bar{\mathcal{C}}$, s and $\bar{s}$, respectively, to a newly defined curve parameter ξ, *i.e.*, $s = h(\xi)$, and $\bar{s} = \bar{h}(\xi)$. When h is invertible, we have $\bar{s} = \bar{h}(h^{-1}(s)) = \bar{h} \circ h^{-1}(s)$, which allows for the use of an alignment function, $g = \bar{h} \circ h^{-1}$, as before. However, when h is not invertible, *i.e.*, when the first particle stops along the first curve for some finite time, g is not defined.

While this formulation allows for a symmetric treatment of the curves, note that a superfluous degree of freedom is introduced, as in [8], because different traversals h and $\bar{h}$ may give rise to the same alignment. While Tagare [8] treats this degree of redundancy by simply searching a larger space of two functions, we remove this additional degree of redundancy by introducing the notion of an *alignment curve*, α, with coordinates h and $\bar{h}$

$$\begin{aligned}\alpha(\xi) \triangleq (h(\xi), \bar{h}(\xi)), \quad \xi \in [0,\tilde{L}], \\ \alpha(0) = (0,0), \quad \alpha(\tilde{L}) = (L,\bar{L}),\end{aligned} \tag{6}$$

where ξ is arc-length along the alignment curve and $\tilde{L}$ is its length. The alignment curve can now be specified by a *single function*, namely, $\psi(\xi)$, $\xi \in [0,\tilde{L}]$, the angle between the tangent to the curve and the x-axis. The coordinates h and $\bar{h}$ can then be obtained by integration

$$\begin{aligned}&h(\xi) = \textstyle\int_0^\xi \cos(\psi(\eta))d\eta, \quad \bar{h}(\xi) = \int_0^\xi \sin(\psi(\eta))d\eta, \\ &\xi \in [0,\tilde{L}].\end{aligned} \tag{7}$$

Note that ψ is constrained by monotonicity ($h' \ge 0$ and $\bar{h}' \ge 0$) to lie in $[0, \frac{\pi}{2}]$. The alignment between $\mathcal{C}$ and $\bar{\mathcal{C}}$ is then fully represented by a single function ψ. The goodness of the match between $\mathcal{C}$ and $\bar{\mathcal{C}}$ can now be rewritten in terms of ψ. First, if

[1] R is a scale constant depending on the average sample size. For all examples in this paper, we set $R = 10$. We have seen experimentally that the alignment is relatively insensitive to the choice of R.

[2] We refer the reader to Basri *et al.* [11] for a discussion of different non-linear norms which were motivated by the extent certain desirable constraints were met. These norms can also be used in the curve alignment framework.

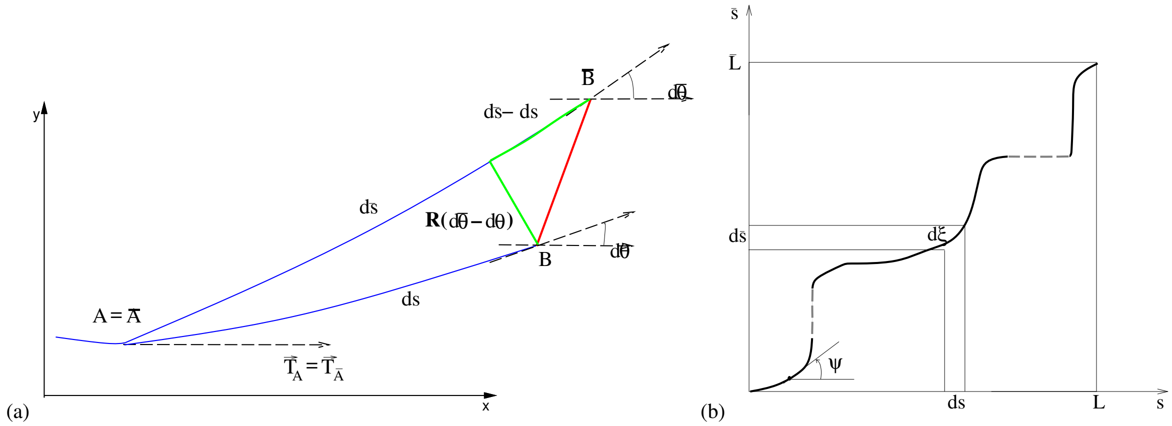


Fig. 1. (a) The cost of deforming an infinitesimal segment AB to segment $\bar{A}\bar{B}$, when the start points and their tangents are aligned ($A = \bar{A}$, $\vec{T}_A = \vec{T}_{\bar{A}}$), is related to the distance $B\bar{B}$, and is defined by $|d\bar{s} - ds| + R|d\bar{\theta} - d\theta|$. (b) The formulation using an alignment curve allows for a finite segment from one curve to be aligned with a single point on one curve, thus allowing for the curve segment deletion or addition.

$h' \neq 0$ and $\bar{h}' \neq 0$ for $\xi \in [\xi_1, \xi_2]$, then $g = \bar{h} \circ h^{-1}$ is well-defined. Using Equations 3 and 4, we get

$$\begin{aligned} \mu[\psi]\big|_{[\xi_1,\xi_2]} &= \int_{\xi_1}^{\xi_2} \left[\left|\frac{dh}{d\xi} - \frac{d\bar{h}}{d\xi}\right| + R\left|\frac{d\theta}{ds}\frac{ds}{d\xi} - \frac{d\bar{\theta}}{d\bar{s}}\frac{d\bar{s}}{d\xi}\right| \right] d\xi \\ &= \int_{\xi_1}^{\xi_2} \Big[|\cos(\psi) - \sin(\psi)| \\ &\qquad + R|\kappa(h)\cos(\psi) - \bar{\kappa}(\bar{h})\sin(\psi)| \Big] d\xi. \end{aligned} \tag{8}$$

Second, consider that one of h' or $\bar{h}'$ is zero at a point, say $h'(\xi) = 0$, implying that $h(\xi)$ maps to a corresponding interval $[\bar{h}(\xi), \bar{h}(\xi + d\xi)]$. This cost of mapping a point to an interval is defined by enforcing continuity of the cost with deformations: consider the cost of mapping the interval $[h(\xi), h(\xi + d\xi)]$ to the interval $[\bar{h}(\xi), \bar{h}(\xi + d\xi)]$ as $d\xi \to 0$, *i.e.*, as $\psi \to \frac{\pi}{2}$, or as $\cos(\psi) \to 0$ in Equation 8. Similarly, the case where an interval in the first curve is mapped to a point in the second curve, is the limiting case of $\psi \to 0$, or $\sin(\psi) \to 0$ in Equation 8. Since both limits are well-defined, using Equation 8 the overall cost of the alignment $\mu[\psi]$ is well-defined in all cases, and the optimal alignment is computed as

$$\begin{cases} \psi^* = \underset{\psi}{\operatorname{argmin}}\, \mu[\psi]\big|_{[0,\tilde{L}]} \\ 0 \leq \psi \leq \frac{\pi}{2}, \ \displaystyle\int_0^{\tilde{L}} \cos(\psi) d\xi = L, \text{ and } \int_0^{\tilde{L}} \sin(\psi) d\xi = \bar{L}. \end{cases} \tag{9}$$

D. Distance Between Curves

In many computer vision applications including object recognition and handwritten character recognition, finding the similarity between curves, or equivalently a dissimilarity distance between them, is a key element. We can use the alignment computed between two curves to define a distance measure between two curves, which is naturally defined from the cost of deformations:

Definition 2: The *edit distance* between two curve segments $\mathcal{C}$ and $\bar{\mathcal{C}}$ is the cost of the optimal alignment of the two curves, $d(\mathcal{C}, \bar{\mathcal{C}}) = \mu(\psi^*)$.

The metric property of a shape distance measure[3] is potentially useful in several applications. For example, in indexing into large databases, symmetry and triangle inequality have been used to efficiently organize the database organization [16], [17], [18], [19].

Proposition 1: The edit distance is a metric, *i.e.*, d satisfies the following properties: (i) $d(\mathcal{C}_1, \mathcal{C}_1) = 0$, (ii) $d(\mathcal{C}_1, \mathcal{C}_2) = d(\mathcal{C}_2, \mathcal{C}_1)$, (iii) $d(\mathcal{C}_1, \mathcal{C}_3) \leq d(\mathcal{C}_1, \mathcal{C}_2) + d(\mathcal{C}_2, \mathcal{C}_3)$.

Proof: (i) and (ii) follow directly from the definition of d. To prove (iii) let ψ_{ij}^*, $i, j = 1, 2, 3$, specify the optimal alignment between C_i and C_j. Let $[s_1, s_1 + ds_1]$ be an arbitrary interval on $\mathcal{C}_1$, let $[s_2, s_2 + ds_2]$ be the corresponding intervals on $\mathcal{C}_2$ under alignment ψ_{12}^*, and let and $[s_3, s_3 + ds_3]$ be the interval corresponding to $[s_2, s_2 + ds_2]$ on $\mathcal{C}_3$, under alignment ψ_{13}^*. Denote the composed correspondence between $\mathcal{C}_1$ and $\mathcal{C}_3$ which takes $[s_1, s_1 + ds_1]$ to $[s_3, s_3 + ds_3]$ as $\hat{\psi}_{13} = \psi_{12}^* \circ \psi_{23}^*$. Using the inequalities $|df_1 - df_3| \leq |df_1 - df_2| + |df_2 - df_3|$, where $f = s, \theta$, we have

$$\begin{aligned} |ds_1 - ds_3| + R|d\theta_1 - d\theta_3| \leq & |ds_1 - ds_2| + R|d\theta_1 - d\theta_2| \\ & + |ds_2 - ds_3| + R|d\theta_2 - d\theta_3|, \end{aligned}$$

which when integrated leads to $\mu[\hat{\psi}_{13}] \leq \mu[\psi_{12}^*] + \mu[\psi_{23}^*]$. Now, using the fact that ψ_{13}^* specifies the optimal alignment of $\mathcal{C}_1$ and $\mathcal{C}_3$, we have $\mu[\psi_{13}^*] \leq \mu[\hat{\psi}_{13}]$ or $\mu[\psi_{13}^*] \leq \mu[\psi_{12}^*] + \mu[\psi_{23}^*]$, which together with the definition of d completes the proposition. ■

Definition 3: The edit distance between two *closed curves* is the minimum cost of the matching the open curve segments starting at any pair of arbitrary points P_1 and $\bar{P}_1$, and terminating there after having traversed the entire curve, *i.e.*, $d(\mathcal{C}_{closed}, \bar{\mathcal{C}}_{closed}) = \min\limits_{[s_1, \bar{s}_1]} d(\mathcal{C}\big|_{[s_1, s_1+L]}, \bar{\mathcal{C}}\big|_{[\bar{s}_1, \bar{s}_1+\bar{L}]})$, where L and $\bar{L}$ are the lengths of the closed curves $\mathcal{C}$ and $\bar{\mathcal{C}}$, respectively.

The computation of edit distance can be sensitive to the choice of parameters as noted in [2] and also to the manner the curves are sampled. Moreover, computing the normalized edit distance is a non-trivial task [20]. The edit distance, how-

[3]There may be instances where human shape comparison does not observe metric properties [14], [15].

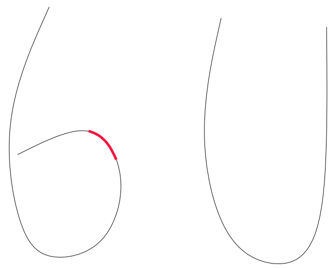


Fig. 2. Some characters differ in curvature only at a few points, and have small edit distance. In this figure "6" and "U", differ in curvature only at the few points that are highlighted.

ever, has the advantage of working well in the presence of articulation and deformation of parts, because of the use of intrinsic properties. This is critical for object recognition. However, for handwritten character recognition this can also present a drawback, Figure 2: edit distance penalizes local differences in lengths and curvatures, and some character pairs like "6" and "U", which differ in curvature only at a few different points, may appear similar. In such cases, global properties must also be taken into account, and the Euclidean distance between corresponding points has been used as a distance measure. The similarity transformation that minimizes the least-squares distance between the corresponding points is found [21], and the average distance between corresponding points is used as the Euclidean distance measure between curves [22], [2]. We will use either edit distance or the Euclidean distance as is appropriate to the underlying application.

III. An Algorithm for Finding the Optimal Alignment Curve

This section describes a dynamic-programming algorithm for finding the optimal alignment curve α^* for any two curves $\mathcal{C}$ and $\bar{\mathcal{C}}$. Dynamic programming has been used for curve matching [23], [24], [3], [2] and deformable-template-based segmentation [25], and in speech recognition [26]. In order to use dynamic programming to find the optimal alignment, the distance function has to satisfy the *optimal substructure* property [27], which we show first.

Proposition 2: The edit distance has *optimal substructure*, $i.e.$, for $\xi_1 < \xi_2 < \xi_3$, $s_i = h^*(\xi_i)$, $\bar{s}_i = \bar{h}^*(\xi_i)$, $i = 1, 2, 3$, we have

$$d(\mathcal{C}\big|_{[s_1,s_3]}, \bar{\mathcal{C}}\big|_{[\bar{s}_1,\bar{s}_3]}) = d(\mathcal{C}\big|_{[s_1,s_2]}, \bar{\mathcal{C}}\big|_{[\bar{s}_1,\bar{s}_2]}) + d(\mathcal{C}\big|_{[s_2,s_3]}, \bar{\mathcal{C}}\big|_{[\bar{s}_2,\bar{s}_3]}). \quad (10)$$

The basic idea is that under the optimal alignment, the alignment of aligned subsegments also has to be optimal. If that is not the case, the non-optimal alignments of the subsegments can be replaced by another to get an alignment with a lower total cost. The detailed proof is omitted due to space constraints [28]. We now describe how the optimal alignment curve α^* is found.

The alignment curve of $\mathcal{C}$ and $\bar{\mathcal{C}}$ is a curve in the 2D plane whose axes are specified by the curve segments $\mathcal{C}$ and $\bar{\mathcal{C}}$. We discretize $\mathcal{C}$ and $\bar{\mathcal{C}}$ at samples $s_1, s_2, \ldots, s_n$ and $\bar{s}_1, \bar{s}_2, \ldots, \bar{s}_m$, respectively, Figure 3a. The alignment curve is then represented by a sequence of N points $(\alpha_1, \ldots, \alpha_N)$, where $\alpha_k = (s_{i_k}, \bar{s}_{j_k})$, $i_k \in \{1, \ldots, n\}$, $j_k \in \{1, \ldots, m\}$, $k = 1, \ldots, N$, and $\alpha_1 = (s_1, \bar{s}_1)$ and $\alpha_N = (s_n, \bar{s}_m)$. That we have constrained the match at each respective endpoints of the two curve segments is not a restriction, since the match also considers *all* possible deletions of segments from the beginning and the end of each curve.

Let $d(i,j)$ denote the cost of matching the discrete curve segments $\mathcal{C}\big|_{[s_1,s_i]}$ and $\bar{\mathcal{C}}\big|_{[\bar{s}_1,\bar{s}_j]}$; let $\delta([k,i],[l,j])$ denote the cost of matching subsegments $\mathcal{C}\big|_{[s_k,s_i]}$ and $\bar{\mathcal{C}}\big|_{[\bar{s}_l,\bar{s}_j]}$. The optimal substructure property of the distance function (Equation 10) allows us to write

$$d(i,j) = \min_{k,l} \left[d(i-k, j-l) + \delta([i-k,i],[j-l,j]) \right], \quad (11)$$

which gives a recipe for computing the edit distance $d(\mathcal{C}, \bar{\mathcal{C}})$ via dynamic programming. The match cost is found by sequentially updating the dynamic-programming table, and the optimal alignment by tracing through the table [27]. We discretize ψ, as a first approximation, to nine values achieved by using a template, Figure 3a. This template limits the choices of k and l, which in turn limits the number of segments that can be merged and then matched to three. We have to compute $d(i,j)$ at every point in the 2D grid. Hence, the complexity of matching curve segments is $O(n^2)$, where n is the number of samples along the curve segments. Note that merging does not increase the complexity, as we allow only a fixed number of segments to merge at each point.

Observe that our proposed approach is analogous to the "edit distance" approach [29] for matching strings, where three operations are allowed: (i) relabeling a character, (ii) deleting a character, and (iii) inserting a character. In the curve alignment domain, the infinitesimal curve segment serves the purpose of a "letter"; stretching and bending of infinitesimal segments correspond to the "relabel edit", removal of an infinitesimal segment of $\mathcal{C}$ is analogous to the "delete edit" and the removal of an infinitesimal segment of $\bar{\mathcal{C}}$ is analogous to the "insert edit". The key difference between string edit distance [29] and our algorithm is that we allow for a merge operation of multiple curve segments. Our algorithm is also similar to the attributed string matching of Tsai and Yu [23]. Other curve matching methods based on dynamic programming [24], [3], [2] also use a similar update step to compute the cost. The main difference in our approach is the use of a fixed template which keeps the complexity at $O(n^2)$.

A. Aligning Closed Curves

The main difference between aligning curve segments and closed curves is that in the former, the start points of the two curves $\mathcal{C}$ and $\bar{\mathcal{C}}$ correspond, $i.e.$, s_1 is mapped to $\bar{s}_1$, while in the latter, the start point correspondence is not known. In order to match closed curves, we need to find the optimal start point correspondence as well. Note that it is sufficient to fix a start point $\bar{s}_1$ on curve $\bar{\mathcal{C}}$, and then find the optimal alignments for all possible start points on the curve $\mathcal{C}$. It is straightforward to see why. Let ψ_1^* be the optimal alignment when we fix $\bar{s}_1$ as the start point on $\bar{\mathcal{C}}$. Let s_1 be the optimal start point on $\mathcal{C}$ and s_2 be the corresponding point of $\bar{s}_2$ under ψ_1^*. Instead of $\bar{s}_1$, if we now choose $\bar{s}_2$ as the start point on $\bar{\mathcal{C}}$, one alignment maps s_2 to $\bar{s}_2$, and s_1 to $\bar{s}_1$. This alignment is same as ψ_1^*, and is the optimal

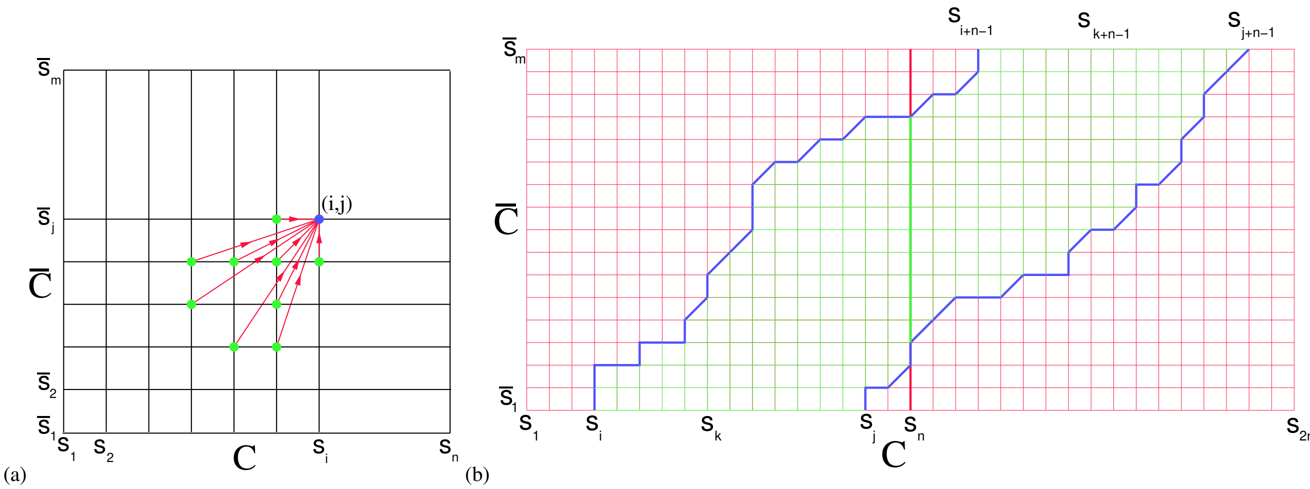


Fig. 3. Left: This figure illustrates the template that is used to find the edit distance of curve segments using dynamic programming. Discrete samples along the curves are the axes. The entry at (i, j) represents $d(i, j)$. To update the cost at (i, j) (blue dot) we limit the choices of the k and l in Equation 11, so that only costs at the a limited set of points (green dots) are considered. Right: This figure illustrates the grid used by the dynamic-programming method to compute the optimal alignment curve for closed curves. The first curve $\mathcal{C}$ is repeated. If the blue curves are optimal alignment curves from s_i and s_j, then the alignment curve from s_k for $i < k < j$ does not cross the blue lines, so the search can be restricted to the green area.

alignment due to Proposition 2. Hence, we fix the start point on curve $\bar{\mathcal{C}}$ and find optimal alignments for all start points on $\mathcal{C}$.

This may appear to increase the computational complexity by the number of sample points along the closed curve to $O(n^3)$, as in previous approaches [24], [3], [2]. Heuristics have been used to reduce the need of having to optimize over n^2 starting point correspondences. For example, Gdalyahu and Weinshall [2] use the distribution of optimal rotation parameters necessary to align the first few points to select plausible start point correspondences. The approaches of Ueda and Suzuki [24] and Milios and Petrakis [3] are not guaranteed to find the optimal alignment for closed curves. As noted in [3], the approach by Ueda and Suzuki [24] may not find the optimal correspondence as only the cost of the best path is stored at each cell in the dynamic-programming table. Milios and Petrakis [3] alleviate this problem by storing the top K options at each cell, but the optimal correspondence is not guaranteed.

The need to find the optimal alignment for each start point correspondence, to ensure that the global minimum of the functional for aligning closed curves is found is a rather restrictive constraint, and motivates the use of the following approach. Observe that the optimal alignment curves corresponding to different start points s_i cannot intersect. Specifically, let α_i^* be the optimal alignment curve from $(s_i, 0)$ to $(s_i + L, \bar{L})$, and α_j^* from $(s_j, 0)$ to $(s_j + L, \bar{L})$. Then, for any $s \in [s_i, s_j]$, the optimal alignment curve from $(s, 0)$ to $(s + L, \bar{L})$ has to lie between α_i^* and α_j^*, and this leads to a significant reduction in the search space, Figure 3b. This is similar to the technique used by Maes [30] for the cyclic string-to-string correction problem. To capture the cyclic nature of the closed curves, the first curve is duplicated in the dynamic-programming grid, Figure 3b, and we compute the optimal alignment curves $\alpha_i^* = (\alpha_{i1}, \ldots, \alpha_{ik}), i = 1, \ldots, n$, where $\alpha_{i1} = (s_i, \bar{s}_1)$, $\alpha_{ik} = (s_{i+n-1}, \bar{s}_m)$, and k is the number of points in α_i^*. When each optimal alignment curve is computed, the search space is split into two, and in analogy to binary search adds a function of $log(n)$ to the overall complexity, for a total $O(n^2 log(n))$ for matching closed curves. Table 4 plots the computational times for curve alignment.

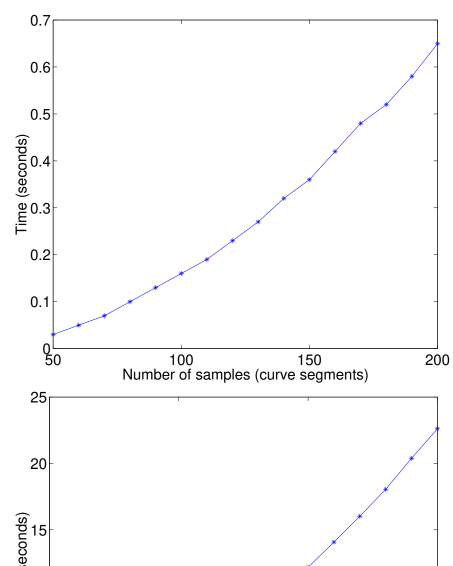


Fig. 4. Running times on an SGI Indigo II (195 MHz) for matching curve segments (top) and closed curves (bottom).

B. Effect of Scale

Note that due to the strict use of intrinsic properties, the proposed curve alignment is invariant to relative rotations and translations of the two curves. However, the dissimilarity measure is not scale-invariant as the stretching term in the functional is not

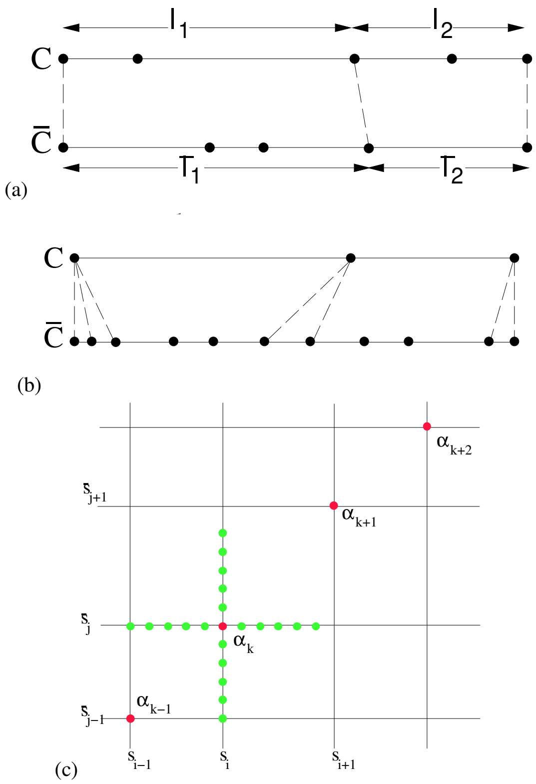


Fig. 5. This figure illustrates the effects of differential sampling. (a) and (b) show the optimal correspondence (dotted lines) for two equal-length line segments (sampled at black dots). Recall that the number of segments that can be merged and then matches is limited to three. When sampling rates are roughly the same the correspondence and the cost are marginally affected (a). When the curves are sampled very differently the optimal correspondence is significantly affected (b). We alleviate the effect of sampling in a post-processing stage (c). The red dots denoted by $\alpha_l, l = k-1, k, k+1, k+2$ are points along the optimal alignment curve. In the post-processing step these alignment curve points are allowed to "relax" along horizontal and vertical lines. In this case, α_k can become any one of the green dots.

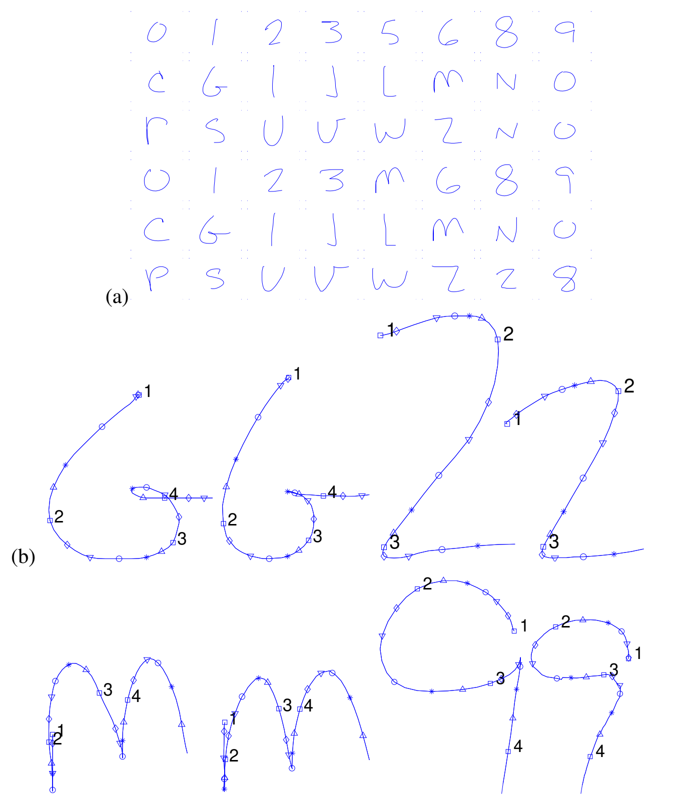


Fig. 6. (a) Sample handwritten characters in our database. (b) Examples of the optimal correspondence. The alignment is indicated by coloring portions of the aligned curves by identical colors with a number indicating the each portion's end point. Observe that the matches are intuitive.

scale-invariant. Methods that rely on using locally scale invariant measures [2] often have difficulty matching curves with the same but largely unequal sized parts. Thus, the determination of scale requires a consideration of the entire curve. This can be done by finding a global optimal scale parameter λ for one curve to match the other and then minimize over several scales. Since this is asymmetric, we will instead scale up one curve by $\sqrt{\lambda}$ and scale down the other curve by the same factor to reach a common curve, giving rise to the functional

$$\int \left|\sqrt{\lambda}ds - \frac{1}{\sqrt{\lambda}}d\bar{s}\right| + R\left|d\theta - d\bar{\theta}\right|. \qquad (12)$$

The optimal scaling factor λ^{opt} is then computed as $\underset{\lambda}{\operatorname{argmin}}\, \mu_\lambda[g]$, which can be computed using gradient descent. Note that only a small range of λ, bounded by the ratios of the largest and smallest sampling intervals, needs to be examined, since the functional is strictly increasing beyond this rather limited range.

C. *Effect of Sampling on Curve Matching*

Our approach is robust to the absolute sampling of the curves, $i.e.$, when sampling rates are doubled or tripled on both curves. However, a differential in the relative sampling rate of aligned curve segments can affect the optimal correspondence and the match cost. Figure 5 illustrates this effect in two cases. In the first case, Figure 5a, the sampling rates are only somewhat different, the correspondence is good and the cost is affected only slightly. The correspondence and hence the cost can be improved by a post-processing step by "fine-tuning" the computed alignment curve points by allowing them to move along the horizontal and vertical grid-lines to minimize the match cost, Figure 5c. In the second case, when the sampling rate is substantially different along corresponding curve segments, the alignment and costs are affected more significantly, Figure 5b. This effect is best handled by resorting to larger templates, requiring additional computational effort. As a practical alternative the curves can be resampled a priori, to ensure that the sampling rates are not drastically different.

IV. Applications of Curve Alignment

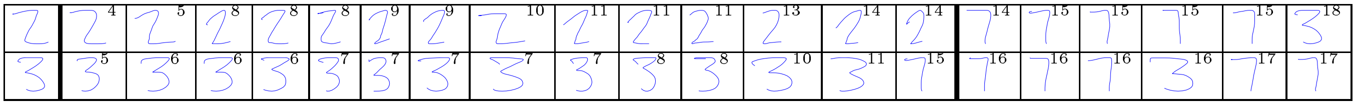
In this section, we illustrate that the proposed curve alignment in fairly generic and useful across many applications.

A. *Handwritten character recognition*

Handwritten character data is typically acquired by sampling the trace of the writer's pen and is represented by a pair of coordinates (x, y), resulting in a curve segment, Figure 6a. Handwritten character data is thus inherently one-dimensional in nature, rendering curve alignment as a natural choice to measure

TABLE I

TOP 20 MATCHES FOR TWO HANDWRITTEN DIGITS.



We used 90 handwritten digits (15x {2,3,6,7,8,9})in this experiment. The number next to the matching character is the Euclidean distance.

character similarity. Figure 6b shows a few examples of the alignment found between two characters using our framework. We use a database of 6 different digits with 15 samples of each for a total of 90 items[4]. Matching is done between each pair of digits in the database. The top 20 matches of a few sample digits are shown in Table I. Note that in most cases (one error only), the top matches are other samples of the same digit. This is a typical result.

Working with a large database necessitates the use of character prototypes [6], [5] to reduce the number of matches required. The prototype can either be a representative sample [5] or an average curve [6]. We use an "average" character as the prototype, Section IV-B. The cost of matching each "unknown" character is computed to all the average characters, and the nearest neighbor is chosen as the correct match. We used a database of 327 digits and alphabets (34 categories) written by one subject. The top five matching prototypes for a few sample characters are shown in Table II. We consider the character to be correctly recognized if the top matching prototype is correct, $i.e.$, we say a "2" is recognized correctly if the top prototype is "2". Out of 327 characters 323 were correctly recognized using the Euclidean measure, leading to a recognition rate of 98.8%. The performance of other methods were significantly lower, Table II.

B. *Curve Averaging and Morphing*

Curve averaging has several applications in computer vision including prototype formation, shape morphing, and computational atlases. The curve alignment framework can be used to compute an average of a set of curves. The average curve is computed by averaging the intrinsic properties of the corresponding curve subsegments. Figure 7 shows the average curve for fish outlines and handwritten digits. Observe that in each case the average curve maintains the general shape as well as the local features and their relationships, and is recognizable as an instance of that category. Note that the average handwritten characters were used as prototypes in the handwritten character recognition experiments of Section IV-A with excellent results.

The proposed curve matching framework can also be used to generate a morphing sequence to "interpolate" between two shapes that are not very dissimilar. The main idea is to find the optimal correspondence between the two shapes, and deform each curve segment to its corresponding segment, using a weighted average of their intrinsic properties. Figure 8 shows examples of morphing a pair of fishes, and a cat to a kangaroo. Observe that all the intermediate shapes are meaningful.

[4]We gratefully acknowledge Jayashree Subrohmania, IBM Pen Technologies Group of Thomas J Watson Research Center for providing us with the handwritten character data.

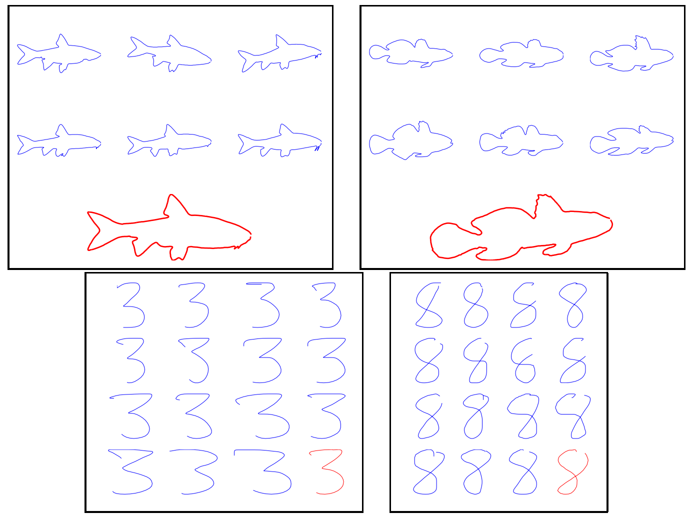


Fig. 7. Original shapes (blue) and their computed averages (red). Observe that in all cases, the average curves are meaningful, and are recognizable as an instance of that category.

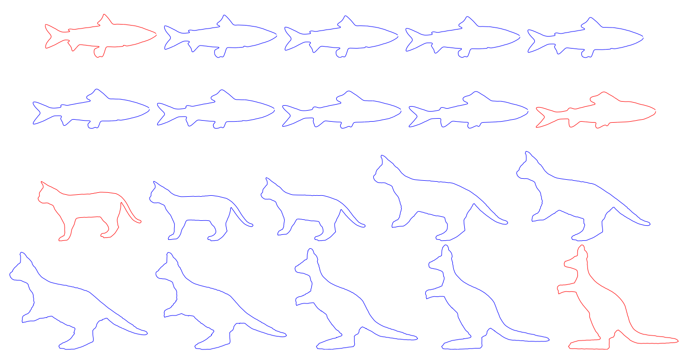


Fig. 8. Two examples of deforming a shape outline to another. The original shapes are shown in red, and the intermediate ones in blue. Observe that the intermediate shapes are meaningful.

C. *Object recognition*

We now examine the effectiveness of curve alignment for matching shapes which are represented by their outlines. For curve alignment to be effective in object recognition, it has to perform well under a variety of visual transformations like occlusion, articulation and deformation of parts, and view point variation. Figure 9 shows that the curve matching algorithm works well in the presence of some of the commonly occurring visual transformations like affine transformations, modest amounts of viewpoint variation and articulation, stretching, and bending of parts. Figure 10 examines the effects of partial occlusion. Curve matching gives the intuitive correspondence in presence of occlusion, when the spatial arrangement of parts is

TABLE II

COMPARISON OF SEVERAL METHODS FOR HANDWRITTEN CHARACTER RECOGNITION.

Method	Errors	% recog.
Euclidean	4	98.8%
Edit	38	88.4%
Hungarian	52	84.1%
ICP	63	80.7%
Softassign	72	69.9%

The top five matching categories for a few sample characters using different distance measures are shown. The number next to each match is the cost of the match in each case. The left table shows results of using the Euclidean distance measure. In the first six rows, the character is correctly recognized, whereas in the last two rows, the top match is incorrect. The table in the middle shows some typical examples where recognition using edit distance, shape contexts using Hungarian assignment [31], ICP [32] and Softassign [33] gives incorrect results. Edit distance under-performed in this task mainly due to the difficulties discussed in Section II-D. shape contexts [31] uses the global features and has problems differentiating between similar characters like "U" and "V". Both ICP [32] and softassign [33] allows partial matches, leading to confusion between characters likes "1" and "L". The table on the right summarizes the recognition rates for all the methods.

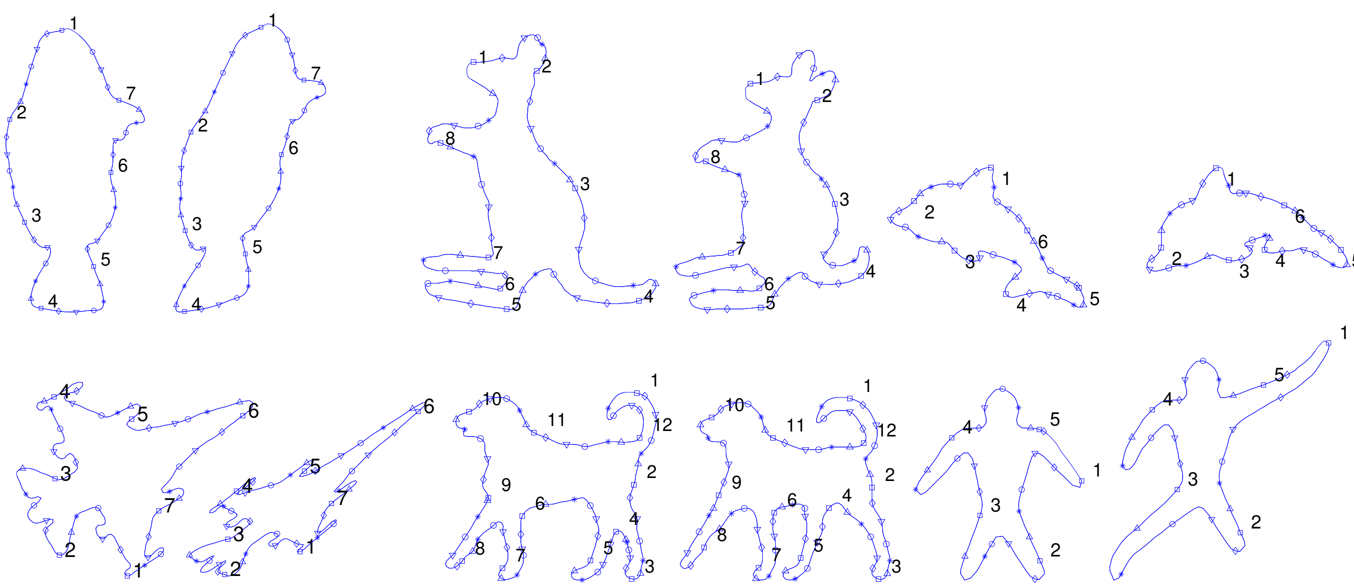


Fig. 9. This figure illustrates the performance of curve alignment in the presence of **affine transformation** (left column), **viewpoint variation** (middle column), and **articulation and deformation of parts** (right column). Curve matching gives the intuitive correspondence in all cases, $i.e.$, salient parts correspond.

not significantly affected by the occlusion, Figure 10 (top row). On the other hand, it may fail when occlusion affects the overall part structure of the shape, Figure 10 (bottom row).

We examine the effectiveness of the proposed edit distance between shape outlines on the database of shapes created for testing the compression rates for MPEG7 [34]. The database consists of 70 categories with 20 shapes in each category for a total of 1400 shapes. The performance is measured by counting the number of correct matches in the top 40 matches [34]. As our approach is not invariant to the "flip" transformation, for each pair of shapes, we compute the optimal alignments for the shapes before and after flipping one of the shapes, and take the minimum of the two as the match cost. The retrieval rate for our approach then is 78.17%. The currently published best performing approaches use curvature scale space [35], correspondence of visual parts [34], and matching shape contexts [36], with retrieval rates of 75.44%, 76.45%, and 76.51%, respectively.

Finally, we have applied our curve alignment method to the GESTURE dataset created by Milios and Petrakis [3], who also provided us with their results and human similarity judgements data. In their paper, they evaluated four methods, namely, Fourier descriptors [37], sequential moments [38], geometric moments [39], and their method based on dynamic programming [3]. Human similarity judgements on the top 50 matches

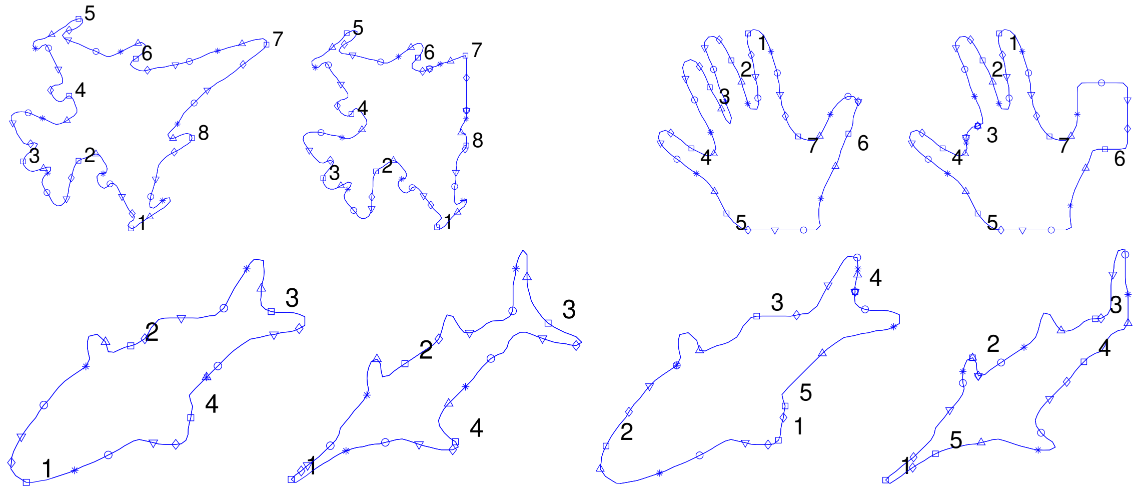


Fig. 10. This figure illustrates the performance of curve alignment in the presence of **partial occlusion**. Top row shows two examples where curve matching gives the intuitive correspondence. Observe that the effects of occlusion is limited to the affected parts. Bottom row illustrates that curve alignment may give the unintuitive correspondence in presence of occlusion that affects the overall part structure. Bottom left: The correspondence between the fishes is intuitive, with the tails, heads and the fins correctly matched. Bottom right: Observe the effect of occluding part of the tail of the fish on the right. The occlusion, though small, affects the part structure of the shape, and gives the wrong correspondence, with the head of the fish on the left matched to a fin on the fish on the right.

TABLE III
THE RETRIEVAL RATES FOR MPEG7 SHAPE DATABASE.

Method	Performance
CSS [35]	75.44%
Visual Parts [34]	76.45%
Shape Contexts [36]	76.51%
Curve Edit Distance	**78.17%**

were used to judge whether a match is correct and evaluated using a precision-recall diagram[5]. We augmented these results by applying shape contexts using Hungarian method [31] and curve alignment to this dataset. The results for all methods are plotted using the precision-recall diagram, Figure 11. Observe that the proposed method significantly outperforms all the competing methods.

A key advantage of the proposed curve alignment for recognition tasks is that it can be used in practical applications due to its high efficiency and excellent recognition rates, even under a moderate range of visual transformations. The main drawback of using this approach when applied to recognition of shapes is that the curve-based representation does not explicitly capture the regional aspect of a shape and is sensitive to the presence and spatial arrangement of parts, Figure 10. Nevertheless, this framework is useful for a variety of applications involving indexing into and organizing a database of shapes.

V. CONCLUSION

We have presented a novel method for finding the optimal correspondence between two curves based on the concept of an *alignment curve*. The optimal alignment is then used to define a degree of dissimilarity between two shapes. We have developed a fast and highly efficient dynamic-programming algorithm for finding the optimal alignment between pairs of curve segments and also closed curves. The exclusive use of intrinsic properties in our formulation leads to invariance under rigid transformation. Our experiments show that curve alignment is robust under a variety of visual transformations. We have illustrated the usefulness of the alignment curve approach for a variety of computer vision applications including handwritten character recognition, prototype formation, shape morphing, and object recognition.

[5]Precision is the percentage of similar shapes retrieved with respect to total number of retrieved shapes. Recall is the percentage of similar shapes retrieved with respect to total number of similar shapes in the database.

ACKNOWLEDGEMENTS

Philip Klein acknowledges the support of NSF Grant CCR-9700146 and Benjamin Kimia acknowledges the support of NSF grants IRI-0083231 and BCS-9980091. We are grateful to Farzin Mokhtarian for providing the fish outlines and Jayashree Subrohmania for the handwritten character data.

REFERENCES

[1] N.J. Ayache and O.D. Faugeras, "HYPER: A new approach for the recognition and positioning of two-dimensional objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 44–54, January 1986.
[2] Yoram Gdalyahu and Daphna Weinshall, "Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 12, pp. 1312–1328, December 1999.
[3] E. Milios and E.G.M. Petrakis, "Shape retrieval based on dynamic programming," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 141–146, 2000.
[4] C.C. Tappert, "Cursive script recognition by elastic matching," *IBM Journal of Research Development*, vol. 26, no. 6, pp. 765–771, 1982.
[5] S. Connell and A.K. Jain, "Learning prototypes for on-line handwritten digits," in *Proceedings of International Conference on Pattern Recognition*, Brisbane, Australia, August 1998, pp. 182–184, Computer Society Press.
[6] B. Wirtz, "Average prototypes for stroke-based signature verification," in *Proceedings of the International Conference on Document Analysis and Recognition*, Ulm, Germany, August 1997, pp. 268–272.
[7] Isaac Cohen, Nicholas Ayache, and Patrick Sulger, "Tracking points on deformable objects using curvature information," in *Proceedings of the European Conference on Computer Vision*, Santa Margherita, Ligure, Italy, 1992, pp. 458–466.

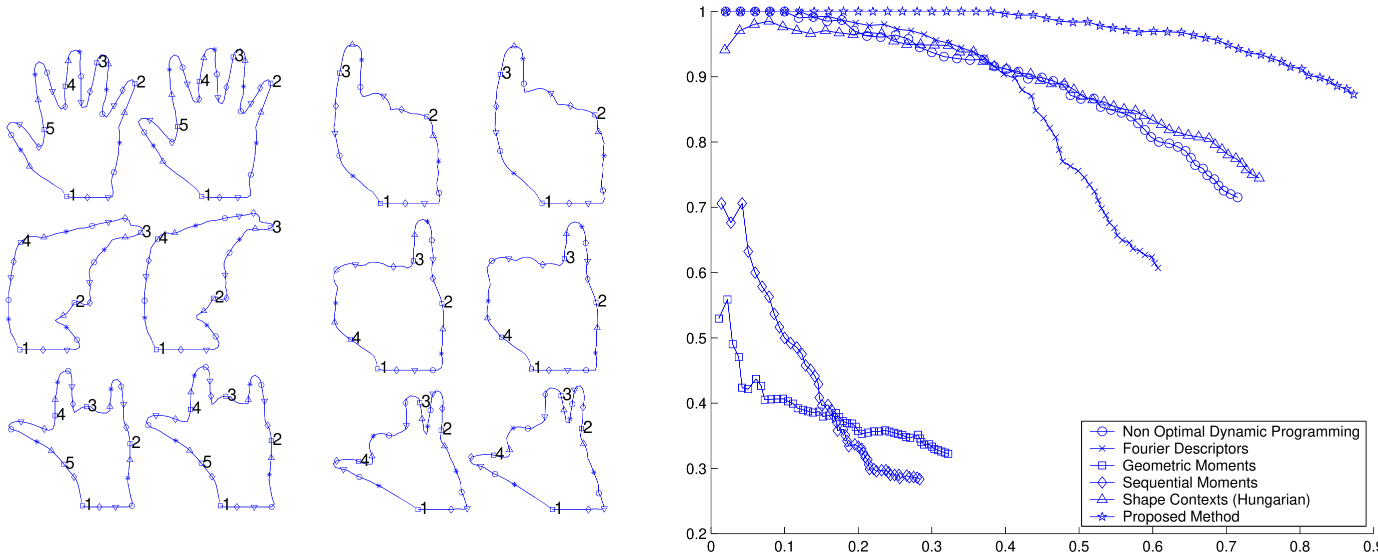


Fig. 11. Left: The result of matching some representative gestures. Observe that the correspondences is intuitive in all cases. Right: Precision-recall diagram for the GESTURE dataset. The results of Fourier descriptors,sequential moments,geometric moments,and non-optimal dynamic programming approaches are from the original paper. We have added two additional results, shape contexts and our technique to this dataset. Observe that the proposed approach outperforms all the competing methods.

[8] Hemant D. Tagare, "Shape-based non-rigid correspondence with application to heart motion analysis," *IEEE Trans. Medical Imaging*, vol. 18, no. 7, pp. 570–578, July 1999.

[9] J.T. Schwartz and M. Sharir, "Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves," *International Journal of Robotics Research*, vol. 6, no. 2, pp. 29–44, 1987.

[10] S. Umeyama, "Parameterized point pattern matching and its application to recognition of object families," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 2, pp. 136–144, February 1993.

[11] Ronen Basri, Luiz Costa, Davi Geiger, and David Jacobs, "Determining the similarity of deformable shapes," *Vision Research*, vol. 38, pp. 2365–2385, 1998.

[12] Laurent Younes, "Computable elastic distance between shapes," *SIAM Journal of Applied Mathematics*, vol. 58, pp. 565–586, 1998.

[13] Michael Kass, Andrew Witkin, and Demetri Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[14] David W. Jacobs, Daphna Weinshall, and Yoram Gdalyahu, "Classification with nonmetric distances: Image retrieval and class representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 583–600, 2000.

[15] A. Tversky, "Features of similarity," *Psychological Review*, vol. 84, no. 4, pp. 327–352, 1977.

[16] Thomas B. Sebastian, Philip N. Klein, and Benjamin B. Kimia, "Shock-based indexing into large shape databases," in *Seventh European Conference on Computer Vision*, Copenhagen, Denmark, May 28-31 2002, pp. Accepted – To Appear, IEEE.

[17] Sergey Brin, "Near neighbor search in large metric spaces," in *Proc. Intl. Conf. on Very Large Databases (VLDB)*, 1995, pp. 574–584.

[18] J. Uhlmann, "Satisfying general proximity/similarity queries with metric trees," *Information Processing Letters*, vol. 40, pp. 175–179, 1991.

[19] P. Yianilos, "Data structures and algorithms for nearest neighbor search in general metric spaces," in *ACM-SIAM Symposium on Discrete Algorithms*, 1993, pp. 311–321.

[20] A. Marzal and E. Vidal, "Computation of normalized edit distances and applications," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, pp. 926–932, 1993.

[21] K.S. Arun, T.S. Huang, and S.D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 5, pp. 698–700, September 1987.

[22] H.C. Liu and M.D. Srinath, "Partial shape classification using contour matching in distance transformation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 11, pp. 1072–1079, November 1990.

[23] W.H. Tsai and S.S. Yu, "Attributed string matching with merging for shape recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 7, no. 4, pp. 453–462, July 1985.

[24] N. Ueda and S. Suzuki, "Learning visual models from shape contours using multiscale convex/concave structure matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 4, pp. 337–352, April 1993.

[25] Hemant D. Tagare, "Deformable 2-D template matching using orthogonal curves," *IEEE Trans. Medical Imaging*, vol. 16, no. 1, pp. 108–117, February 1997.

[26] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[27] Richard E. Bellman and Stuart E. Dreyfus, *Applied Dynamic Programming*, Princeton University Press, 1962.

[28] Thomas Sebastian, Philip Klein, and Benjamin Kimia, "Curve matching using alignment curve," Tech. Rep. LEMS 184, LEMS, Brown University, June 2000.

[29] R.A. Wagner and M.J. Fischer, "The string-to-string correction problem," *Journal of the Association for Computing Machinery*, vol. 21, pp. 168–173, 1974.

[30] Maurice Maes, "On a cyclic string-to-string correction problem," *Information Processing Letters*, vol. 35, pp. 73–78, June 1990.

[31] Serge Belongie and Jitendra Malik, "Matching with shape contexts," in *IEEE Workshop on Content-based Access of Image and Video Libraries*, 2000.

[32] Paul J. Besl and Neil D. McKay, "A method for registration of 3D shapes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, February 1992.

[33] Anand Rangarajan, Haili Chui, and Fred L. Bookstein, "The softassign Procrustes matching algorithm," in *Information Processing in Medical Imaging*. 1997, pp. 29–42, Springer.

[34] L. J. Latecki, R. Lakamper, and U. Eckhardt, "Shape descriptors for non-rigid shapes with a single closed contour," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR00)*, 2000, pp. 424–429.

[35] Farzin Mokhtarian, Sadegh Abbasi, and Joseph Kittler, "Efficient and robust shape indexing through curvature scale space," in *International Workshop on Image Databases and Multimedia Search*, Amsterdam, The Netherlands, 1996, pp. 35–42.

[36] S. Belongie, J. Malik, and J. Puzicha, "Matching shapes," in *Proc. Intl. Conf. Computer Vision (ICCV01)*, 2001, pp. I: 454–461.

[37] T.P. Wallace and P.A. Wintz, "An efficient three-dimensional aircraft recognition algorithm using fourier descriptors," *Computer Graphics and Image Processing*, vol. 13, no. 1, pp. 99–126, May 1980.

[38] L. Gupta and M.D. Srinath, "Contour sequence moments for the classification of closed planar shapes," *Pattern Recognition*, vol. 20, pp. 267–272, 1987.

[39] M.K. Hu, "Visual pattern recognition by moment invariants," *IEEE Trans. Information Theory*, vol. 8, no. 2, pp. 179–187, February 1962.