Representation and Self-Similarity of Shapes

Davi Geiger, Tyng-Luh Liu, Robert V. Kohn

Abstract— **Representing shapes in a compact and informative form is a significant problem for vision systems that must recognize or classify objects. We describe a compact representation model for two dimensional (2-D) shapes by investigating their self-similarities, and constructing their shape axis trees (SA-trees). Our approach can be formulated as a variational one (or equivalently as MAP estimation of a Markov random field). We start with a 2-D shape, its boundary contour, and two different parameterizations for the contour (one parameterization is oriented counter-clockwise and the other clockwise). To measure its self-similarity the two parameterizations are matched to derive the best set of one-to-one point-to-point correspondences along the contour. The cost functional used in the matching may vary, and is determined by the adopted self-similarity criteria, e.g., co-circularity, distance variation, parallelism, and region homogeneity. The loci of middle points of the pairing contour points yield the shape axis, and they can be grouped into a unique free tree structure, the SA-tree. By implicitly encoding the (local and global) shape information into an SA-tree, a variety of vision tasks, e.g, shape recognition, comparison and retrieval can be performed in a more robust and efficient way via various tree-based algorithms. A dynamic programming algorithm gives the optimal solution in $O(N^4)$, where N is the size of the contour.**

Keywords— **Shape representation, self-similarity, variational matching, dynamic programming, MRF**

I. Introduction

An effective and compact shape representation system is a critical element for various computer vision application. Perhaps the most explicit way to represent a shape in 2-D is by a set of pixels contained within the shape. Alternatively one can describe a shape by its boundary contour(s), which is typically a more compact representation than the straightforward region representation. There are many other possible representations for shapes and we seek a compact one. However, we can not just naively invoke the idea of the Kolmogorov complexity[1], as the goal must include the construction of an efficiently computable system. Today, no methodology exists to search for "the Kolmogorov idea."

If shapes are represented with global descriptors, then articulated objects tend to be dissimilar since these changes/deformations may change the global appearance of objects considerably while the entire deformation is concentrated on specific points. Alternatively, if shapes are described by local boundary descriptors, e.g., [2], then they do not account for region information, object parts, and global properties such as symmetries (see Fig. 1). An ideal representation system of shapes should not be sensitive to small changes of appearance; otherwise similar objects of the same class or partially occluded will be represented very differently.

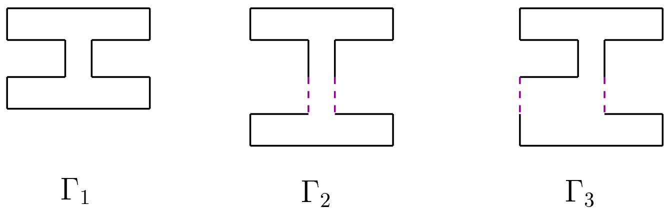


Fig. 1. Γ_2 and Γ_3 are derived from Γ_1 by the same amount of stretches at different locations. A shape descriptor accounting for symmetries can favor Γ_2 over Γ_3 when comparing to Γ_1.

Our aim is to develop a *shape axis tree* (SA-tree) representation of objects, where the graph edges of an SA-tree correspond to the structure of object parts (see Fig. 2). In particular, we are interested in shape representations that are

(i) *complete*, i.e., all the object information is stored in the representation.
(ii) *simple*, *meaningful*, and *compact*, i.e., redundancies are removed by capturing symmetries, region information, articulations (local deformations), and dividing the object into "object parts" with as few parts as needed for any level of specified details (via parameter setting).
(iii) *stable*, i.e., robust under small variations of the shape (including noise variations and articulations).
(iv) *easily computable*, i.e., existence of polynomial time algorithms (on the size of the shape) to find its representation.

A. Shape Representation

Our formulation is a variational one. It can also be interpreted as the MAP estimate of a Markov random field (MRF). Starting with a two dimensional shape and its boundary contour, we seek a self-similarity measure to structure the shape representation. The idea is to generate two different parameterizations for the boundary contour and to measure its self-similarity by matching the two parameterizations, i.e., by matching pairs of contour points (and their tangents) along the boundary contour. Depending on the class of shapes and human perception factors, the matching (self-similarity) criterion may vary, e.g., co-circularity, parallelism, and region homogeneity. The optimal matching will give a set of pairing points for all the boundary contour. This representation, the set of pairing of all contour points, can be used to retrieve the original contour as chain of points in a straightforward way.

The set of middle points of the optimal pairing contour points gives a shape axis. A shape axis can bifurcate yield-

D. Geiger and R. V. Kohn are with the Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, NY 10012. E-mail: geiger@cims.nyu.edu, kohn@cims.nyu.edu

T.-L. Liu is with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan. E-mail: liutyng@iis.sinica.edu.tw

[1]Kolmogorov Complexity of an object (i.e., a binary string) is the length of the shortest computer program that runs on a computer and outputs that object.

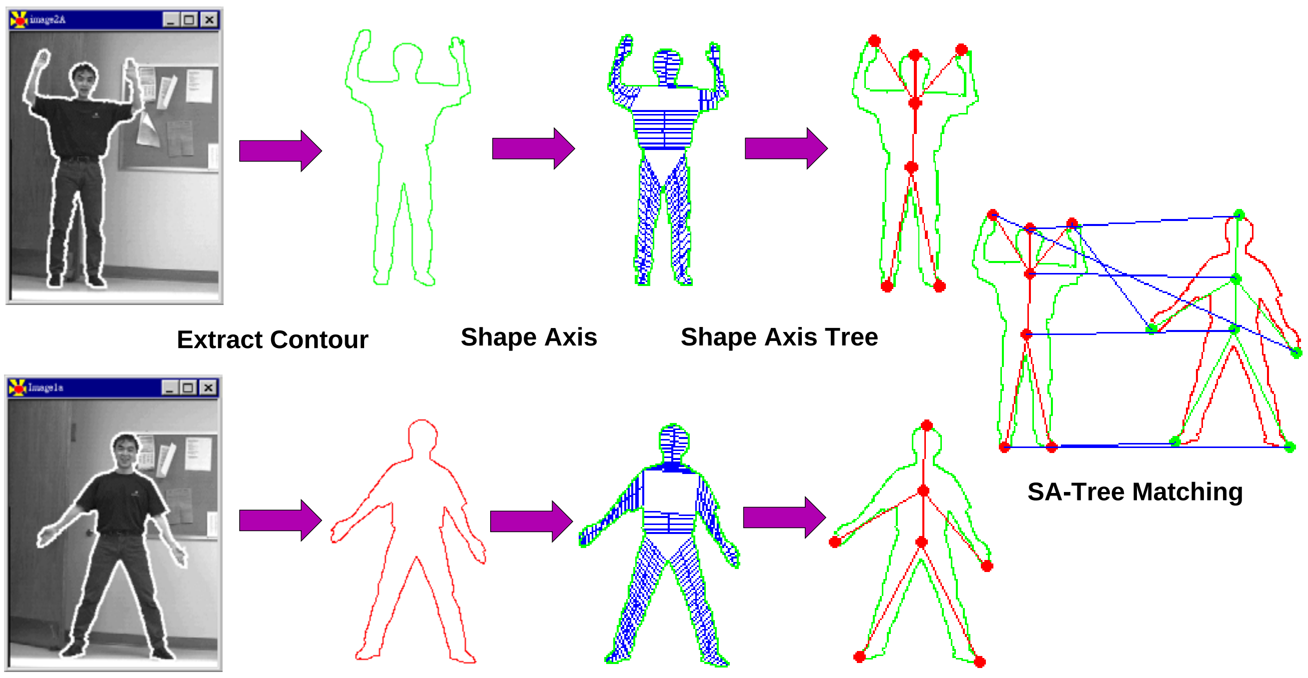


Fig. 2. Constructing the SA-trees of two shape contours extracted from real images. Then, similarity comparison can be performed using tree matching scheme.

ing the notion of structure parts (see Fig. 2). When the optimization criterion is based on mirror symmetry, or equivalently the co-circularity criterion, the shape axis is closely related to the symmetry axis or skeleton.

From the self matching, and pairing of all boundary contour points, we extract a shape axis tree. In an SA-tree, every node corresponds to either an end point (a match of a point with itself) or a bifurcation of the shape axis. The edges of the tree correspond to the object parts.

A.1 Open Shapes

Our method can also be applied to construct shape axes from open shape contours. In this case the starting point for each parameterization is one of the extreme points (ends) of the open contour, no longer the same one, each parameterization starts from the opposite extreme (end) point. While the shape axis for an open contour can be derived by essentially the same algorithm (designed for closed contours), it is possible that its corresponding SA-tree may degenerate into an SA-forest (see Fig. 10e).

B. Previous Work and Comparison

Blum first proposed to represent shapes by their symmetry and thickness [4]. Binford [3] is also a pioneer and brought the attention to generalized cylinders. Other early work includes [1], [18]. In [8], [9], affine transformations are addressed, and direct 3-D considerations were given in [26]. Pizer et al. have proposed a computational model for object representation via "cores", or regions of high medalists in intensity images [5]. Ogniewicz [19] presented an efficient method based on Voronoi diagrams. Leymarie and Levine [15] have used the grassfire transform (from the psychology literature). In [29], Segen has described a *layered graph* representation for 2-D shapes, where the relations between object parts are encoded in the vertices. Siddiqi and Kimia's work [31] has an interesting mathematical formulation, preserving the intuition of the grassfire idea. It is a development of a reaction-diffusion equation where the symmetry axis is obtained and described by the development of shocks (first, second, third, and fourth order ones). It is motivated by the framework for shape analysis via shock-diffusion equations [14]. Following this line, Siddiqi et at. have addressed the 2-D shape matching problem via comparing *shock graphs* [33]. Tari et al. [34] have defined an *edge strength function* where its level curves corresponding to the smoothed propagating glassware front. The function can be applied to determine the axes of local symmetries for 2-D shapes, and recently the approach has be extended to shapes of arbitrary dimensions [35]. Sclaroff and Pentland [28] proposed a modal matching method for correspondence and recognition based on describing the generalized symmetries of shapes. They used finite element techniques to transform shapes into a modal space and to address shape similarity by modal deformation energy. Zhu and Yuille's work [39] as an effort to use the symmetry axis representation to recognize and represent flexible objects from the silhouettes.

Our view differs from previous work in that we are seeking a variational optimization approach for the problem.

II. Variational Matching of Curves

We shall determine the shape axis of a curve by finding a "good match" between the curve and its mirror image. It is convenient to begin, however, with the more general

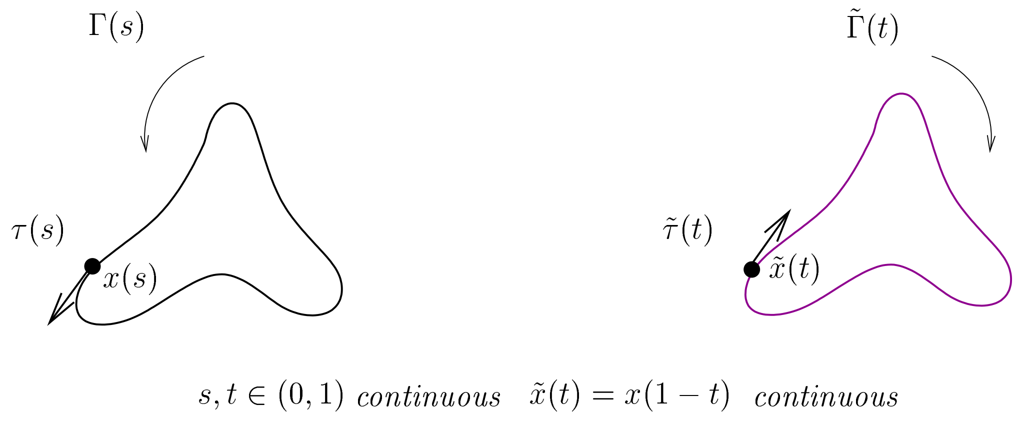


Fig. 3. The contour $\Gamma(s)$ and its mirror version $\tilde{\Gamma}(t)$. $\tau(s)$ and $\tilde{\tau}(t)$ are unit tangents.

task of finding a "good match" between a pair of curves in the plane.

Consider a pair of parameterized curves $\Gamma = \{x(s) : 0 \le s \le 1\,, x(0) = x(1)\}$ and $\tilde{\Gamma} = \{\tilde{x}(t) : 0 \le t \le 1\,, \tilde{x}(0) = \tilde{x}(1)\}$. By a *match* between Γ and $\tilde{\Gamma}$ we mean a monotone correspondence between the two curves, taking endpoints to endpoints. A match can be represented in several different ways. One representation uses the map $s \mapsto t = t(s)$ such that $x(s)$ corresponds to $\tilde{x}(t(s))$. A second uses the inverse map $t \mapsto s = s(t)$, so $\tilde{x}(t)$ corresponds to $x(s(t))$. A third alternative has the advantage of treating the two curves symmetrically: it specifies a *pair* of monotone functions $s(\sigma)$ and $t(\sigma)$, each defined for $0 \le \sigma \le 1$, such that $x(s(\sigma))$ corresponds with $\tilde{x}(t(\sigma))$. (The first two alternatives are special cases, obtained by taking $s(\sigma) = \sigma$ and $t(\sigma) = \sigma$.)

To determine a "good match" we solve an appropriate variational problem. Concentrating for the moment on our third, more symmetrical representation, the matching energy should have the form

$$\int_0^1 F(x, \tau; \tilde{x}, \tilde{\tau}; s', t')\, d\sigma, \tag{1}$$

where $\tau = x_s/|x_s|$ and $\tilde{\tau} = \tilde{x}_t/|\tilde{x}_t|$ are the oriented unit tangent vectors to Γ and $\tilde{\Gamma}$ at $x(s)$ and $\tilde{x}(t)$. Our notation in (1) is somewhat abbreviated: the integrand must be evaluated at the appropriate point $x = x(s(\sigma))$, $\tau = \tau(s(\sigma))$, $\ldots$, $s' = ds/d\sigma$, and $t' = dt/d\sigma$.

Our framework imposes two structural conditions on the matching energy density F. The first is the symmetry condition

$$F(p, \xi; q, \eta; v, w) = F(q, \eta; p, \xi; w, v),$$

which assures that the notion of "matching" is symmetric, i.e., the two curves Γ and $\tilde{\Gamma}$ play equivalent roles. The second is the scaling condition

$$F(p, \xi; q, \eta; \lambda v, \lambda w) = \lambda F(p, \xi; q, \eta; v, w) \quad \text{for all } \lambda > 0,$$

which makes the energy invariant under change-of-variable in σ:

$$\begin{aligned}
& \int F\left(x, \tau; \tilde{x}, \tilde{\tau}; \frac{ds}{d\sigma}, \frac{dt}{d\sigma}\right) d\sigma \\
= & \int F\left(x, \tau; \tilde{x}, \tilde{\tau}; \frac{ds}{d\sigma}, \frac{dt}{d\sigma}\right) \frac{d\sigma}{d\bar{\sigma}}\, d\bar{\sigma} \\
= & \int F\left(x, \tau; \tilde{x}, \tilde{\tau}; \frac{ds}{d\sigma}\frac{d\sigma}{d\bar{\sigma}}, \frac{dt}{d\sigma}\frac{d\sigma}{d\bar{\sigma}}\right) d\bar{\sigma} \\
= & \int F\left(x, \tau; \tilde{x}, \tilde{\tau}; \frac{ds}{d\bar{\sigma}}, \frac{dt}{d\bar{\sigma}}\right) d\bar{\sigma}.
\end{aligned}$$

This assures that the energy depends only on the match, i.e., the correspondence between $s = s(\sigma)$ and $t = t(\sigma)$, and not on the specific choice of maps $\sigma \mapsto s(\sigma)$ and $\sigma \mapsto t(\sigma)$.

We want our notion of "match" to be geometric, so it is natural to require invariance under translation and rotation:

$$F(p, \xi; q, \eta; v, w) \quad \text{depends on } p,\, q \text{ only through } p - q\,,$$

and

$$F(Rp, R\xi; Rq, R\eta; v, w) = F(p, \xi; q, \eta; v, w)\,,$$

for every orientation-preserving rotation R. These assure that applying the same rigid motion to Γ and $\tilde{\Gamma}$ leaves the energy of their match unchanged. Invariance under scaling is too much to ask, but it is natural to ask that

$$F(\lambda p, \xi; \lambda q, \eta; v, w) = g(\lambda) F(p, \xi; q, \eta; v, w) \quad \text{for } \lambda > 0,$$

for some function $g(\lambda)$. Thus scaling Γ and $\tilde{\Gamma}$ by a common factor λ changes the energy of match by a known amount $g(\lambda)$.

These restrictions leave still considerable freedom. The particular choice of F should depend, of course, on the type of match one seeks.

A. Matching a Curve with Itself

We turn now to our real goal – determining the self-similarity of a closed curve Γ. Locally, our notion of "good

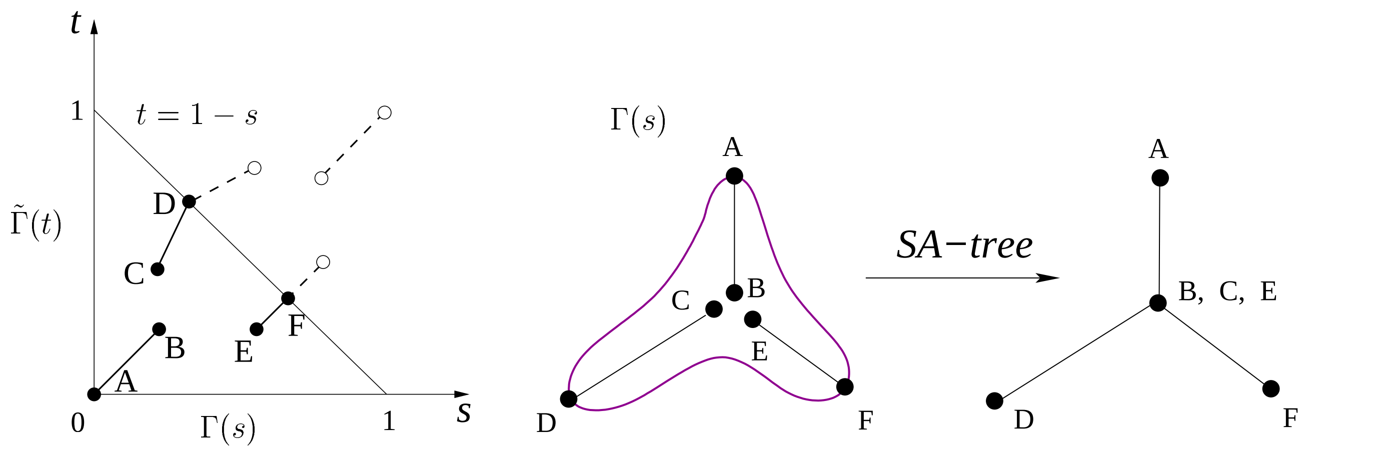


Fig. 4. From left to right: The matching space ($s \times t$) and the graph (where $\mu(s,t) = 1$). The dashed lines represent mirror copies of the solid ones, with respect to the mirror line: $t = 1 - s$. An SA-tree is then obtained.

match" is the one introduced above. Globally, however, the situation is different because we now permit discontinuities. The matches considered in this section are only *piecewise* continuous and monotone.

To explain our new notion of "match" more precisely, consider a simple closed curve Γ in the plane, parameterized monotonically by $x(s)$, $0 \le s \le 1$, with $x(0) = x(1)$. Extending $x(s)$ periodically, we may consider it to be defined for all $s \in \mathbf{R}$, with $x(s+1) = x(s)$. (It is equivalent to think of Γ as a mapping from the unit circle $\mathbf{S}^1$ to $\mathbf{R}^2$.) Let $\tilde{\Gamma}$ be "the same curve traced backwards," parameterized by $\tilde{x}(t) = x(1-t)$ (see Fig. 3). Notice that $\tilde{x}$ is again defined for all t, periodic with period 1.

A match between Γ and $\tilde{\Gamma}$ can be represented as a binary function $\mu(s,t)$:

$$\mu(s,t) = \begin{cases} 1, & \text{if } x(s) \text{ corresponds to } \tilde{x}(t) = x(1-t), \\ 0, & \text{otherwise.} \end{cases}$$

We visualize the match by plotting the points in the s,t plane where $\mu(s,t) = 1$ (see Fig. 4).

There are some structural conditions every match must satisfy. First, to respect the periodicity of $x(s)$ we require

$$\mu(s,t) = \mu(s+1,t) = \mu(s,t+1) \quad \text{for all } s,t. \tag{2}$$

Second, to make Γ and $\tilde{\Gamma}$ play equivalent roles we require that

$$\begin{gathered} x(s) \text{ corresponds with } \tilde{x}(t) = x(1-t) \\ \Updownarrow \\ x(1-t) \text{ corresponds with } \tilde{x}(1-s) = x(s). \end{gathered}$$

This amounts to

$$\mu(s,t) = \mu(1-t, 1-s). \tag{3}$$

Thus, the plot of μ must be symmetric about the line $t = 1 - s$. It is frequently convenient to view the plot of μ as the graph of a piecewise monotone function $t(s)$. Then (3) becomes the condition

$$1 - s = t(1 - t(s))\,. \qquad \textbf{[Mirror Property]}$$

Third, the correspondence must be monotone and continuous except for finitely many jumps, and aside from the jumps every point on Γ must have a unique correspondent on $\tilde{\Gamma}$. In other words, the plot of μ, restricted to the unit square $[0,1] \times [0.1]$, must consist of finitely many monotone graphs; moreover it should cover each axis exactly once (except for jumps) (see Fig. 4).

Every match μ determines a collection of proposed shape axis, by the rule

$$\begin{aligned} \mu(s,t) = 1 &\Longleftrightarrow x(s) \text{corresponds to } \tilde{x}(t) = x(1-t) \\ &\Longleftrightarrow \frac{x(s) + x(1-t)}{2} \quad \text{belongs to a shape axis.} \end{aligned}$$

Where the plot of μ is discontinuous, so is the associated shape axis. We call such discontinuities *bifurcations* of the shape axis. Where the plot of μ crosses the line $t = 1 - s$ the points $x(s)$ and $x(1-t)$ are identical and the associated shape axis meets Γ. We call these points the *leaves* of the shape axis.

Of course, to detect the real symmetries of the figure we cannot use just any match – we must use a good one. Each continuous portion of the match is assigned an energy by the analysis of the last section. The total energy is obtained by adding these contributions, then adjoining "jump energies" associated with the discontinuities. More precisely, we add to (1) an energy term

$$\sum_{\text{jumps}} \texttt{JumpCost}\,,$$

which is interpreted as follows: at a jump, the values $t(\sigma+) = \lim_{\delta \to 0^+} t(\sigma + \delta)$ and $t(\sigma-) = \lim_{\delta \to 0^+} t(\sigma - \delta)$ are different. The associated jump cost can be a function of $|t(\sigma+) - t(\sigma-)|$. In practice we have used a constant cost for jumps. Our "jumps" amount to the vertical discontinuities one sees in the plot of μ (see Fig. 4). There is no need to handle the horizontal jumps separately because they are in one-to-one correspondence with the vertical ones, on account of the symmetry (3). Shapes with bifurcations of

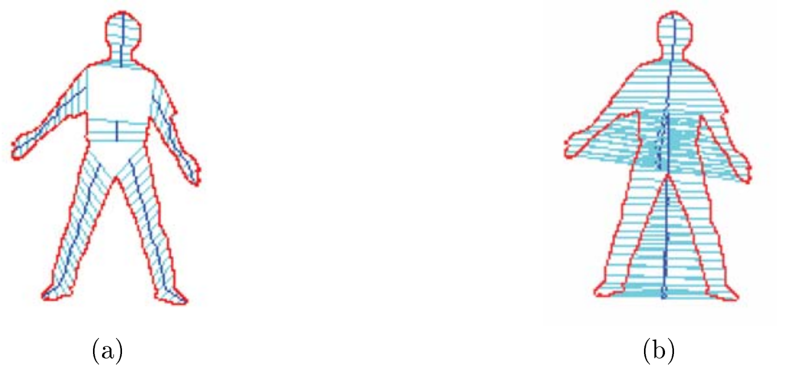


Fig. 5. There are no restriction to the loci of the shape axis where they can be partially inside and partially outside the shape. (a) The distance from the axis to the boundary is small, but bifurcations must be placed. (b) There is no bifurcation, but the distance from the axis to the boundaries tends to be large. The optimal solution can be (a) or (b), depending on the weight of the bifurcation cost (`JumpCost`).

various degrees are considered by our model (see Fig. 8) and the number of vertical discontinuities gives the degree of the bifurcation. One may consider to have the "jump energies" depend directly on the number of bifurcations not the degrees of bifurcations. However, when the cost for a bifurcation does not depend on the degree of bifurcation the following problem may occur: small perturbations near a bifurcation may easily create new branches (parts), that is, increase the degree of the bifurcation without any penalty. We choose to penalize the degree of a bifurcation. In this way, the number of bifurcations is controlled indirectly, and it leads to a more stable system. When analyzing the dynamic programming solution we will investigate this issue thoroughly.

B. Similarity Criteria

Various similarity/symmetry criteria can be considered, e.g., co-circularity, parallelism, region homogeneity.

B.1 Co-Circularity or Mirror Symmetry

We want $F(p,\xi;q,\eta;v,w)$ to favor

$$(q-p)\perp(\xi+\eta) \quad \text{and} \quad (q-p)\|(\xi-\eta)\,.$$

In other words F favors *co-circularity* – the existence of a circle passing through p and q with tangents ξ at p and η at q (see Fig. 6b).

B.2 Stretching and Distance:

It is also natural to favor $v = w$, so that there are no stretchings, and to favor $p = q$ so that elements are matched when they are close (see Fig. 5). Two possible choices of F's are

$$\begin{aligned} F^{(1)}(p,\xi;q,\eta;v,w) &= [(q-p)\cdot(\xi+\eta)]^2\,(v+w) \\ &+ \left[(q-p)\cdot(\xi-\eta)^{\perp}\right]^2(v+w) \\ &+ c\,|v-w|\,|p-q|^2 \end{aligned}$$

and

$$\begin{aligned} F^{(2)}(p,\xi;q,\eta;v,w) &= \frac{[(q-p)\cdot(\xi v+\eta w)]^2}{v+w} \\ &+ \frac{\left[(q-p)\cdot(\xi v-\eta w)^{\perp}\right]^2}{v+w} \\ &+ c\,\frac{|p-q|^2\,|v-w|^2}{v+w}, \end{aligned}$$

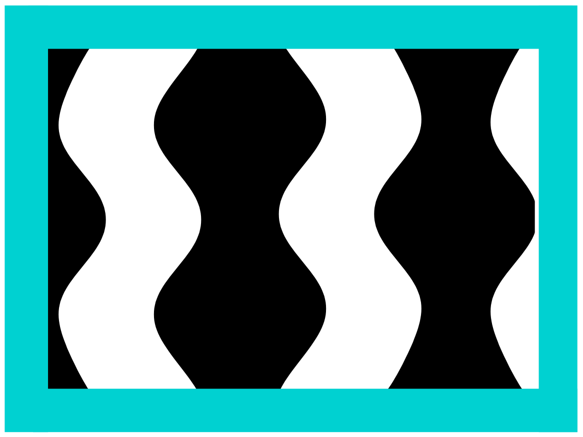
where c is a positive constant. We chose the latter form, somewhat arbitrarily (from previous work [2]), for the examples in this paper. This gives an energy of the form

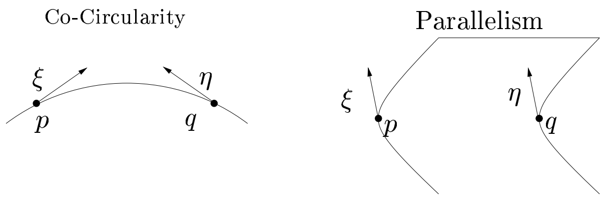
$$\begin{aligned} &E[t(\sigma),s(\sigma)] \\ &= \int_0^1 \Big\{ \frac{[(x(s)-\tilde{x}(t))\cdot(\tau(s)s'(\sigma)+\tilde{\tau}(t)t'(\sigma))]^2}{s'(\sigma)+t'(\sigma)} \\ &+ \frac{\left[(x(s)-\tilde{x}(t))\cdot(\tau(s)s'(\sigma)-\tilde{\tau}(t)t'(\sigma))^{\perp}\right]^2}{s'(\sigma)+t'(\sigma)} \\ &+ c\,|x(s)-\tilde{x}(t)|^2\frac{|s'(\sigma)-t'(\sigma)|^2}{s'(\sigma)+t'(\sigma)} \Big\}\,d\sigma\,, \end{aligned} \qquad (4)$$

where $\tau(s)$ and $\tilde{\tau}(t)$ are the unit tangent vectors at $x(s)$ and $\tilde{x}(t)$ respectively.

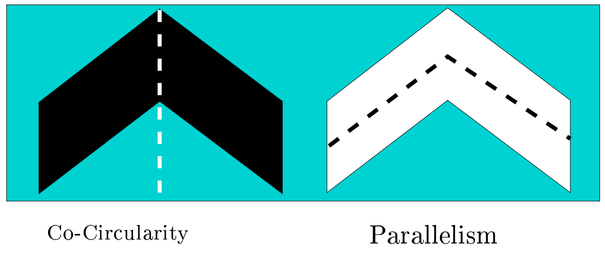
B.3 Parallelism and Distance Variation

Other criteria may also be favorable, and translation (or parallelism) is an interesting one frequently to be considered (see Fig. 6a). It is clear that for a straight segment to be a translation (parallel) of another one, the tangents must be equal. To guarantee a symmetric cost (and account for stretching) one model for parallelism is given by the energy $|\tau(s)s'(\sigma)-\tilde{\tau}(t)t'(\sigma)|$. This criterion is very similar as penalizing for variations of the distance of the matched points along the parameterization. More precisely, we take $\left|\frac{d}{d\sigma}(x(s(\sigma))-\tilde{x}(t(\sigma)))\right| = \left|\,|x_s(s)|\tau(s)s'(\sigma) - |\tilde{x}_t(t)|\tilde{\tau}(t(s))t'(\sigma)\,\right|$. For a pure translation of segments both criteria give zero cost. Distance variations may be important to distinguish two shapes like in Fig. 7, that would have identical self matching energy


(a)



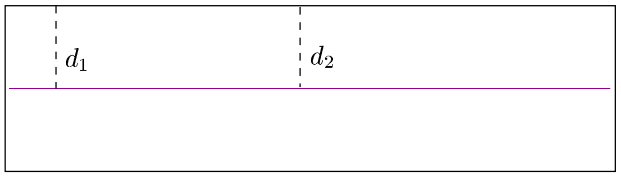
(b)



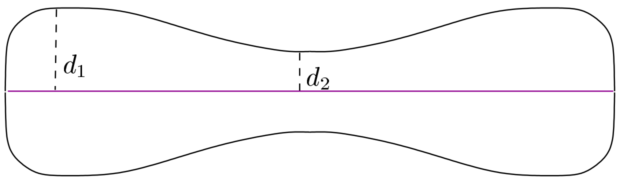
(c)

Fig. 6. (a) It is known, from the Gestalt school, that parallelism (or distance variation) can be more salient than co-circularity as the white region can pop up. (b) The co-circularity criterion is equivalent to mirror symmetry and can be expressed by the relations $(q-p)\perp(\xi+\eta)$ and $(q-p)\|(\xi-\eta)$. The parallelism (or translation) criterion can be expressed as $\xi\|\eta$. It is very similar to a distance variation criterion (see text). (c) Two different shape axes of a same shape based on co-circularity and parallelism, respectively.

otherwise. Distance variation may also be necessary as it accounts for perceptual phenomena (see Fig. 6). A study to choose the preferred human perception model is beyond the scope of this paper.



(a) Distance d is constant



(b) Distance d is varying

Fig. 7. Both figures have perfect mirror symmetry and yield zero cost according to the energy (4). They can only be distinguished by a distance variation criterion, or similar ones like a parallelism criterion.

B.4 Discretization

In practice the curve Γ is given discretely, as a sequence of N points. The data determine $x(s)$ at $s \in \{s_i = i\Delta s;\ i = 0,\dots,N\}$, with $\Delta s = 1/N$ (notice that $x(s_0) = x(s_N)$), and if necessary $x(s)$ is determined at other points by interpolation. For the mirror curve, we have $\tilde{x}(t_j) = x(s_{N-j})$, for $j = 0,\dots,N$. A match between $x(s_i)$ and $\tilde{x}(t_j)$ is defined by $\mu_{ij} = 1$, and μ_{ij} is zero otherwise. We discretize the parameterization $\sigma \in \{\sigma_k = k\Delta\sigma;\ k = 0,\dots,N\}$. It is convenient to set $\Delta\sigma$ to $1/N$, the same step size as Δs and Δt. A match is then defined by the map from $\sigma_k \mapsto (s(\sigma_k), t(\sigma_k))$. The equivalence between both methods is given by $\sigma_k \leftrightarrow \mu_{ij} = 1 \ \rightarrow\ (s_i = s(\sigma_k), t_j = t(\sigma_k))$.

Discretizing the functional (4) and including the cost for jumps we obtain the discrete energy (cost)

$$\begin{aligned} &E[t(\sigma), s(\sigma)] \\ &\approx \sum_{k=0}^{N-1} \Big[(1-\theta(\sigma_k))\,\hat{F}(\sigma_k,\sigma_{k+1})\,\Delta\sigma + \theta(\sigma_k)\times \texttt{JumpCost}\Big] \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\Big[(1-\theta(\sigma_k))\,\hat{F}(\sigma_k,\sigma_{k+1})\Big] + \sum_{\text{jumps}} \texttt{JumpCost}, \end{aligned} \tag{5}$$

where $\hat{F}(\sigma_k,\sigma_{k+1})$ is an abbreviated notation that

$$\begin{aligned} &\hat{F}(\sigma_k,\sigma_{k+1}) \\ &\equiv \hat{F}(s(\sigma_k), t(\sigma_k), s(\sigma_{k+1}), t(\sigma_{k+1})) \\ &= \frac{[(x(s(\sigma_k)) - \tilde{x}(t(\sigma_k)))(\tau(s(\sigma_k))s'(\sigma_k) + \tilde{\tau}(t(\sigma_k))t'(\sigma_k))]^2}{s'(\sigma_k)+t'(\sigma_k)} \\ &+ \frac{\left[(x(s(\sigma_k)) - \tilde{x}(t(\sigma_k)))(\tau(s(\sigma_k))s'(\sigma_k) - \tilde{\tau}(t(\sigma_k))t'(\sigma_k))^{\perp}\right]^2}{s'(\sigma_k)+t'(\sigma_k)} \\ &+ c|x(s(\sigma_k)) - \tilde{x}(t(\sigma_k))|^2 \frac{|s'(\sigma_k) - t'(\sigma_k)|^2}{s'(\sigma_k)+t'(\sigma_k)}, \end{aligned}$$

$s'(\sigma_k) = \frac{s(\sigma_{k+1})-s(\sigma_k)}{1/N}$, and $t'(\sigma_k) = \frac{t(\sigma_{k+1})-t(\sigma_k)}{1/N}$. The indicator function $\theta(\sigma_k)$ is defined to be 1 at σ_k if a jump occurred, i.e., $t(\sigma_k+) \neq t(\sigma_k-)$ (as discussed before, it suffices to look at only the vertical jumps), and 0 otherwise. Analogously,

$$E[\{\mu\}] \approx \frac{1}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left[\mu_{ij}(1-\theta(\mu_{ij})) \cdot \hat{F}(s_i,t_j,s_{i'},t_{j'})\right] + \sum_{\text{jumps}} \texttt{JumpCost}, \tag{6}$$

where θ is again an indicator function for jumps, and we have assumed that $\sigma_k \leftrightarrow \mu_{ij}=1$ and $s_{i'}=s(\sigma_{k+1}), t_{j'}=t(\sigma_{k+1})$.

C. Markov Random Field (MRF) Interpretation

The computational problem is to obtain the optimal set of pairing $(s_i=s(\sigma_k), t_j=t(\sigma_k))$ represented by minimizing μ_{ij} (6). We can interpret the matches as a binary random variables and define a coupled Markov random field distribution for μ_{ij} as

$$P(\mu) = \frac{1}{Z}e^{-E[\{\mu\}]} = \frac{1}{Z}\prod_{i,j=0}^{N-1} e^{-\frac{1}{N}\left[\mu_{ij}(1-\theta(\mu_{ij}))\hat{F}(s_i,t_j,s_{i'},t_{j'})\right]-\theta(\mu_{ij})\texttt{JumpCost}}, \tag{7}$$

where Z is a normalization constant. The lower the energy is the more likely μ's are to be the solution. The MAP estimate will yield the minimum energy. By writing the problem as an MRF we emphasize the local properties of the energy function as well as its global behavior.

III. Self-Matching and SA-Tree via Dynamic Programming

Our task is now to search in the graph $\{s_i \times t_j;\, i,j = 0,1,...,N\}$ for an optimal path $\{\mu^*_{ij}; i,j=0,...,N\}$ that gives the minimum energy (6) under the constraints discussed for the path. We show that it can be computed with dynamic programming in polynomial time, with complexity $O(N^4)$.

Notice that although in most of our illustrations of shape axis, the solution curves are plotted starting from the origin (e.g., see Fig. 4) , i.e., $x(0)$ is matched to $\tilde{x}(0)$, this correspondence is not assumed and not required in our method, i.e., a solution curve does not have to start from the origin.

Single Source Shortest Path (SSSP) or Dijkstra's algorithm. Because the costs of the edges in the graph are always positive (thus the energy function is a sum of positive terms), we have also applied the SSSP algorithm to obtain further efficiency. We will not elaborate the approach based on Dijkstra's algorithm but concentrate on the dynamic programming solution, for simplicity. The SSSP can be readily adapted as described in [6].

A. Dynamic Programming

From the mirror property stated in (3), we shall focus on the lower triangular part of the diagram $\{s_i \times t_j\}$. Any possible match, including the optimal one, can be obtained by searching and scanning downward along diagonal lines, starting from the mirror line $t = 1-s$. The mirror line represents "self matches", i.e., matches between points along the shape contour with themselves. Each diagonal line can be indexed by a that varies over $\{0,1,...,N\}$ and represented as $t=\frac{a}{N}-s$. So, $a=N$ corresponds to the mirror line and $a=0$ corresponds to the origin (s_0,t_0) (see Fig. 8). For convenience, we switch from the representation of matching points $(i,j)\to(s_i,t_j)$ (or $\mu_{ij}=1$) to the representation $[i,a]\to(s_i,\frac{a}{N}-s_i)$ (or $\mu_{ia}=1$).

Dynamic programming sets an initial cost along the mirror line $a=N$ and iterates on $a=N-2,...,0$ (note that $a=N-1$ does not need to be considered since it will cause a path to reach the mirror line at a non-lattice point, i.e., we can simply set the costs to infinity at the line $a=N-1$). Along each diagonal line a it visits every $(s_i,\frac{a}{N}-s_i)$, $i=0,...,a$, solving the subproblem "What is the cost of the best path passing through $(s_i,\frac{a}{N}-s_i)$?".

We use the matrix $\texttt{Cost}[i,a]$ to store the best cost of each subproblem at $(s_i,\frac{a}{N}-s_i)$. With this representation, the second index gives the index a of the diagonal line and the first index fixes the position on the diagonal line a. A sketch of the algorithm is as follows.

A.1 Initialization

Initially, all $\texttt{Cost}[i,a]$'s are set to ∞ except those on the mirror line, i.e., $\texttt{Cost}[i,N]=0$, for $i=0,...,N$. We could instead put a bias for points along the contour that their curvatures are local maxima/minima as they are more likely to be self-matches.

A.2 Iterations on a

Suppose we have solved the subproblems up to line $a+1$, i.e., $\texttt{Cost}[j,b]$, $j=0,1,...,b$ and $b=N,N-1,...,a+1$, are computed. Then, $\texttt{Cost}[i,a], i=0,...,a$ of diagonal line a can be derived by

$$\texttt{Cost}[i,a] \;=\; \min\{G_S[i,a], G_J[i,a]\}\ , \tag{8}$$

where

$$G_S[i,a] = \min_{[j,b]\in D_S[i,a]}\left\{\hat{F}(s_i,t_{a-i},s_j,t_{b-j}) + \texttt{Cost}[j,b]\right\}\ ,$$

$$G_J[i,a] = \min_{[i,b]\in D_J[i,a]}\Big\{\texttt{Cost}[i,b] + \texttt{Cost}[N-b+i,N-b+a] + (3-2\times J[i,b] - 2\times J[N-b+i,N-b+a])\times\texttt{JumpCost}\Big\}, \tag{9}$$

and

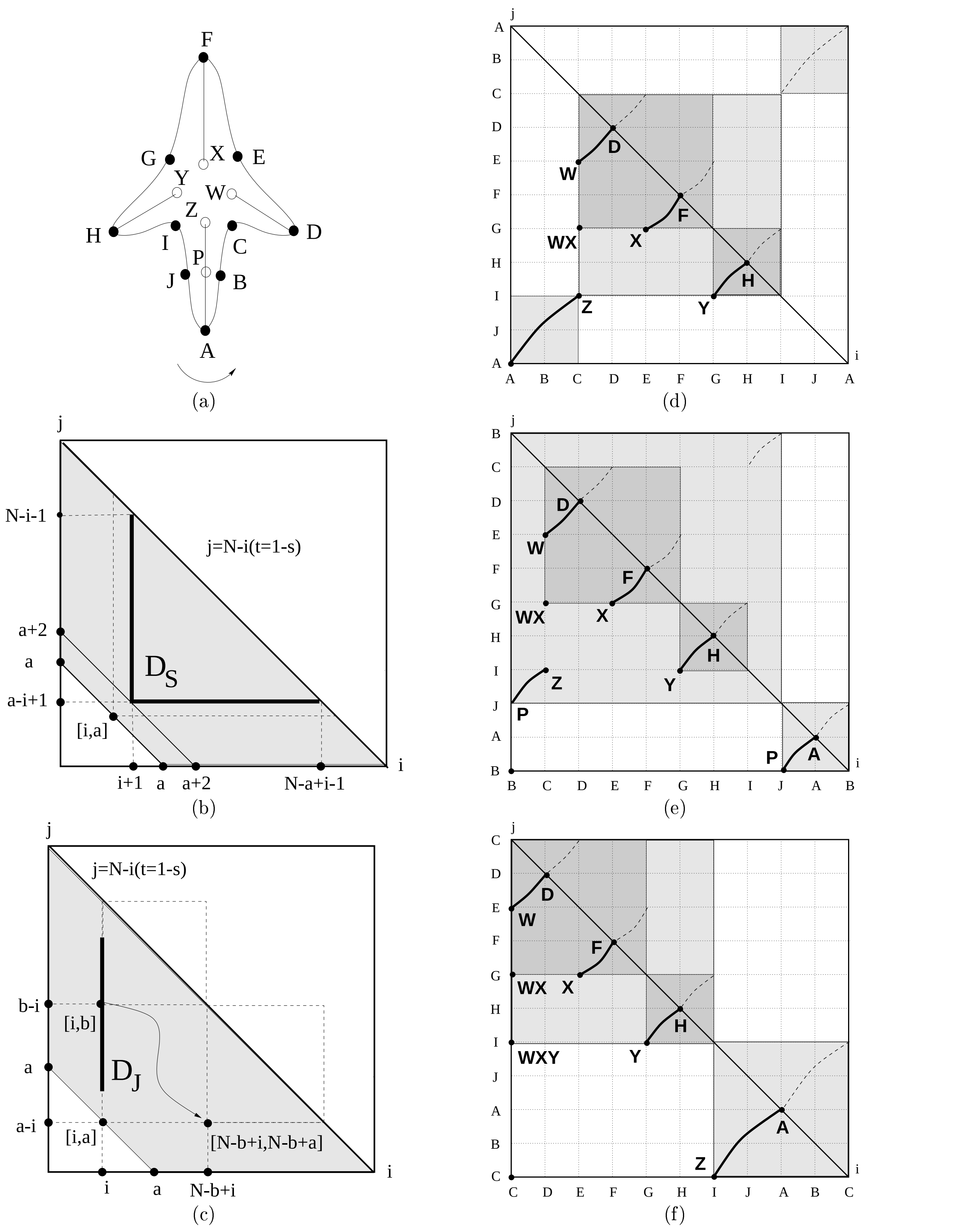


Fig. 8. (a) A shape contour with a degree 4 bifurcation. (b),(c) The subproblem being solved is $(s_i, \frac{a}{N} - s_i)$, i.e., to find `Cost`$[i, a]$. The candidate predecessors can either come from D_S to form a smooth transition or from D_J to form a bifurcation where the grey areas represent subproblems that have been solved. Notice that to form a degree 3 bifurcation with $[i, a]$, we only need to search the domain D_J since for each $[i, b] \in D_J$ the third point, $[N - b + i, N - b + a]$, will be automatically derived. (d),(e),(f) Three different scenarios representing the same solution: the piecewise smooth solutions display a "periodic" property as the starting point on the contour has been set to A, B and C, respectively.

$$\begin{aligned}
J[i,a] &= \begin{cases} 1, & \text{if } \texttt{Cost}[i,a] = G_J[i,a], \\ 0, & \text{if } \texttt{Cost}[i,a] = G_S[i,a], \end{cases} \\
D_S[i,a] &= \{[i+1,d] : \ a+1 < d <= N\} \\
&\quad \bigcup \{[i+k, a+1+k] : \ 0 < k < N-a\}, \\
D_J[i,a] &= \{[i,d] : \ a+1 < d < N-1\} \ .
\end{aligned}$$

The notation G_J represents the cost of grouping the two branches to be part of a bifurcation (due to jumps) and G_S represents the cost of having the predecessor in a smooth path (continuation). D_S and D_J are the domains where dynamic programming searches for possible predecessors to form a smooth path or a bifurcation, respectively (see Fig. 8b, 8c). The binary matrix $J[i,a]$ defined above is to guarantee that each (vertical) jump is penalized only once, and the penalty term in (9) makes sure a degree n bifurcation to be charged `JumpCost` exactly n times.

In order to backtrack the optimal solution, the algorithm also keeps the predecessors

$$\texttt{Back}[i,a] = \begin{cases} \{[j^*, b^*]\}, \quad \text{if } \texttt{Cost}[i,a] = G_S[i,a], \\ \\ \{[i, b^*], [N-b^*+i, N-b^*+a]\}, \\ \text{if } \texttt{Cost}[i,a] = G_J[i,a]. \end{cases} \qquad (10)$$

A.3 Optimal solution and backtracking

In the end this iterative process, from $a = N$ to $a = 0$, will provide $\texttt{Cost}[0,0]$, the total cost of the optimal match. The final stage, $a = 0$, requires special attention due to the random selection of the starting point of the contour.

There are two possibilities: (i) if $x(s_0)$ matches itself, then there is only one predecessor connecting a smooth path to reach the origin, and so $\texttt{Cost}[0,0] = G_S[0,0]$ (see Fig. 8d), and (ii) if $x(s_0)$ does not match itself, then there are two predecessors, say $[0,b]$ and $[N-b, N-b]$. Unlike the usual "jump" case, it does not always imply a bifurcation, due to the periodicity of a closed curve. This is when both predecessors are smooth continuations, i.e., $J[0,b] = J[N-b, N-b] = 0$, and no `JumpCost` should be added to $G_J[0,0]$ (see Fig. 8e). All other cases correspond to forming a bifurcation, and they will be treated as before except that the penalty should be $2 \times (1 - J[0,b] - J[N-b, N-b]) \times \texttt{JumpCost}$. So, when backtracking the optimal match with $\texttt{Cost}[0,0] = G_J[0,0]$, the origin does not represent any match, but only the end of the algorithm. Thus, our method does not require any assumption on the starting point of a given shape.

Complexity. There are $O(N^2)$ subproblems to be solved (evaluating $\texttt{Cost}[i,a]$ for $i = 0, ..., a$ and $a = N-1, ..., 0$), and the complexity of each subproblem is $O(N^2)$ (the size of total searching spaces D_S and D_J is $O(N)$ and the computation complexity of $\hat{F}(s_i, t_{a-i}, s_k, t_{d-k})$ is also $O(N)$, assuming uniform matching). Therefore, the complexity of this algorithm is $O(N^4)$. Backtracking only adds a $O(N)$ cost to the $O(N^4)$ cost of running the algorithm.

B. Analysis of Bifurcations of Degree n

Our analysis originates from a pivotal observation that by examining bifurcations of degree 3 at each stage, one can indeed handle bifurcations of any degree (see Fig. 9).

Let us first concentrate on a shape with a bifurcation of degree 4, see Fig. 8a, and show how our method can handle it. Then, it follows that the general case to deal with bifurcations of any degree is also valid by induction.

Observe that the degree of a bifurcation is the same as the number of smooth branches being grouped together. In Fig. 8e, the four branches in the upper left shaded boxes are merged to form a degree 4 bifurcation $WXYZ$ (again, only need to concentrate on region below the mirror line). So, one may consider the number of branches inside a box as its "degree". Then, we can say that the $WXYZ$ in Fig. 8f is completed by a merging, at the final stage $a = 0$, among the origin and the two shaded boxes of degree 3 and 1, respectively, and it can be denoted as $4 = 3+1+0$, corresponding to $WXYZ = WXY + Z + O$ (here O represents the origin). Similarly, in Fig. 8d and 8e, each degree 4 bifurcation is derived by $4 = 2+1+1$ (or $WXYZ = WX + Y + Z$).

Notice that there are other possible ways to derive $WXYZ$, however they all require the same cost. This is due to the property that each jump is exactly penalized once and a degree n bifurcation will be charged `JumpCost` n times. In equation (9), the penalty term of $G_J[i,a]$ is formulated to maintain this property. Each time three branches being grouped together, one should first pay $3 \times \texttt{JumpCost}$ then check if $2 \times (J[i,b] + J[N-b+i, N-b+a]) \times \texttt{JumpCost}$ can be credited back or not. (The credits account for the considerations that a predecessor can not be a completion of a bifurcation and a jump should be charged only once.) One can verify that in Fig. 8d, 8e and 8f, the total penalty for the degree 4 bifurcation in each case is $4 \times \texttt{JumpCost}$.

C. Backtracking and SA-trees

By starting at the origin $[0,0]$, the value $\texttt{Back}[i,a]$ will take at each step to the predecessor(s). It is either only one predecessor or two, depending on if it is a continuation or a bifurcation. (For simplicity, we will not discuss the special case when $[0,0]$ has two predecessors but still a continuation as in Fig. 8e.) When there is one predecessor it continues accumulating (retrieving) the segment path. When it gives two predecessors we can group them together and argue it has reached a bifurcation node of the tree. To complete a bifurcation node, backtracking should be applied to each of the two predecessors in parallel, and such processes are carried out recursively until reaching points with only one predecessor. Then all these 1-predecessor points will be grouped together to complete the bifurcation node, and the backtracking process starting again from each of them. For example, in Fig. 8d, once the backtracking reaches Z, it starts to collect the 1-predecessor points associated with the underlying bifurcation node. So, W, X and Y are grouped together to form the $WXYZ$ bifurcation node. The backtracking then continue to proceed from each of the W, X, and Y.

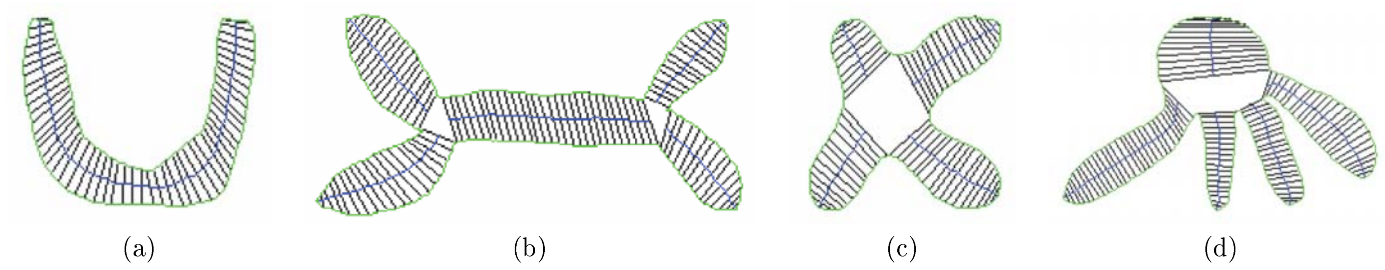


Fig. 9. The algorithm can handle any number of bifurcations of any finite degree. We show examples of (a) none, (b) two degree-3 (c) one degree-4, (d) one degree-5 bifurcations.

Eventually all the backtracking processes will reach the mirror line, i.e., when all the branches reach the mirror line the process is finished. The end points of each of the branches are turned into leaves of the tree. (If $[0, 0]$ only has one predecessor, it is also a leaf.) These nodes correspond to self-matches of points on the contour and so we may refer to them as *contact points* of the match. The nodes of an SA-tree can be classified as

1. Bifurcation nodes: The nodes constructed along the backtracking process correspond to bifurcations of the SA and therefore we call them bifurcation nodes of the SA-tree. We refer to the SA-tree by T. The bifurcations nodes are then referred by *Bifurcation(T)*.
2. Leaf nodes: The nodes created at the end of the backtracking process, where all the branches of the backtracking reaches the mirror line (and possibly the origin), are called leaf nodes. They are the leaves of the tree, and correspond to contact points, i.e., matches between points of the contour with themselves. These are places where the SA meets the shape contour. In the tree structure they correspond to the leaves and we refer to these points by *Leaf(T)*.

Object parts. We have now a language for defining object parts, where every two consecutive nodes of the SA-tree and the edge in between correspond to an object part. How much our definition can describe object parts in the sense of human perception of object parts is still subject of further investigation.

IV. Experimental Results and Application

We have carried out a variety of experiments to test some specific properties of the model, and to display the general quality of the model/approach. In all our experiments, we have used JumpCost = 100 for all the experiments (and the results are robust with respect to small variations), unless specified differently. It takes roughly 4.5 minutes to find the optimal shape axis for a shape contour of 1000 points size, on a Pentium-III 650 Mhz PC.

A. *Shape-Axis Detection*

We have carried out a variety of experiments to test some specific properties of the model, and to display the general quality of the model/approach. In all our experiments, we have used JumpCost = 100 for all the experiments (and the results are robust with respect to small variations), unless specified differently. It takes roughly 4.5 minutes to find the optimal shape axis for a shape contour of 1000 points size.

First, we show an experiment to test how the algorithm is sensitive to perturbations on the shape contour. In Fig. 10a, it shows that when the perturbation becomes significant then it turns into a part and causes bifurcation in the SA. The parameter JumpCost controls the breaking. The larger the JumpCost is the larger a perturbation has to be to cause a new part. The other experiments (Fig. 10b and 10c) are to support our observation that to control the number of bifurcations indirectly via penalizing the degree of a bifurcation leads to a more stable system.

Second, we examine the assertion that our approach can have an SA that is inside and outside the shape contour. Moreover, we analyze the bias our model of SA has towards small distance to the shape contour (given by the terms $|x(s) - \tilde{x}(t)|^2$). Fig. 5 shows how the SA weights the number of bifurcations over the distance term, as the JumpCost vary from a 100 to 1000. It can produce SA's that are outside and inside the shape.

Third, as the main advantage of our method, the algorithm can obtain SA for a 2-D shape with any number of bifurcation of any finite degree. The results for closed and open shapes are shown in Fig. 9 and Fig. 11, respectively.

Finally, various SA's are computed for a variety of shape contours as shown in Fig. 13.

A.1 Open shapes and SA-Forest

The proposed dynamic solution can be extended to extract the SA's from open contours. Experimental results for open shape contours are shown in Fig. 11. For an open contour, it has two distinct end points, and they start the two parameterizations accordingly. To find the shape axis, we only need to modify the final step solving for the subproblem $(0, 0)$. Notice that if a shape axis is derived by a match where the two end points are not matched to each other, the corresponding SA-tree will degenerate into an SA-forest (see Fig. 11e).

B. *An Application to Shape Comparison*

We now demonstrate the efficiency of the proposed model, and to show that with a good representation, it is possible to have a general framework to address the issues of shape stretching, articulation, and occlusion simultaneously.

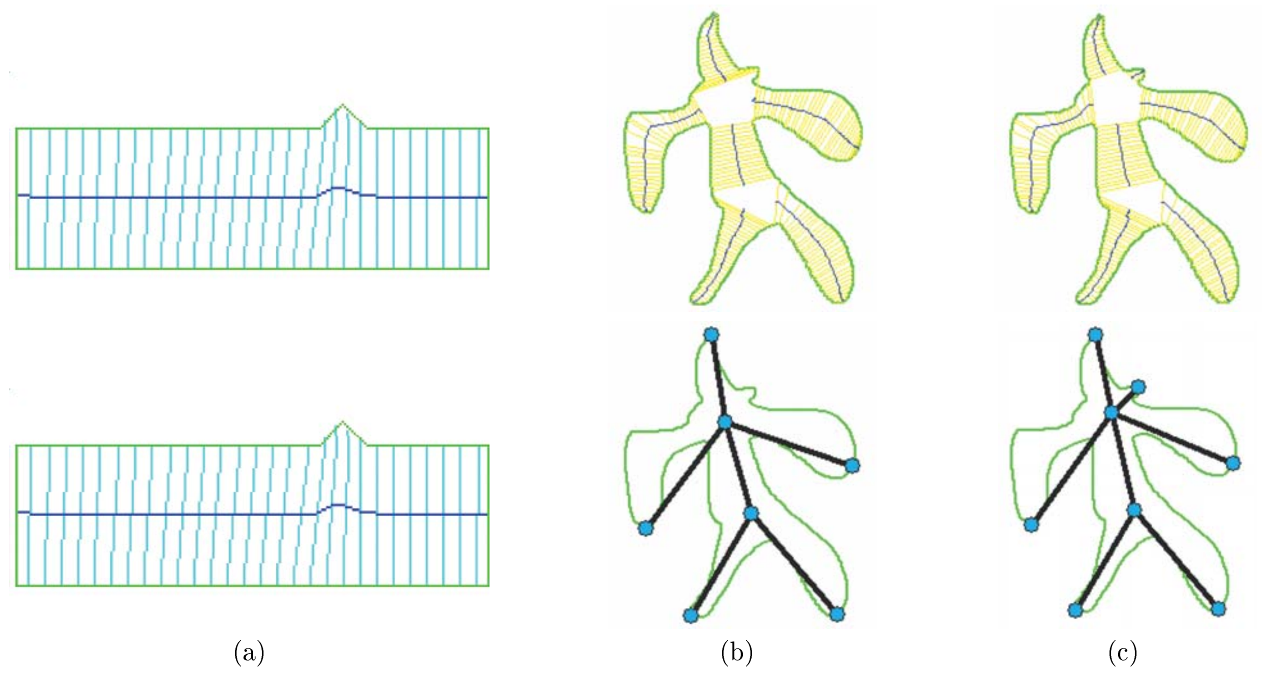


Fig. 10. (a)Rectangular shape with small perturbation and with large perturbation. The shape will break into an additional new part if the perturbation is large enough. (b) To control the number of bifurcations indirectly via penalizing the degrees of bifurcations is more stable,compared to the results in (c), derived when `JumpCost` depends directly on the number of bifurcations.

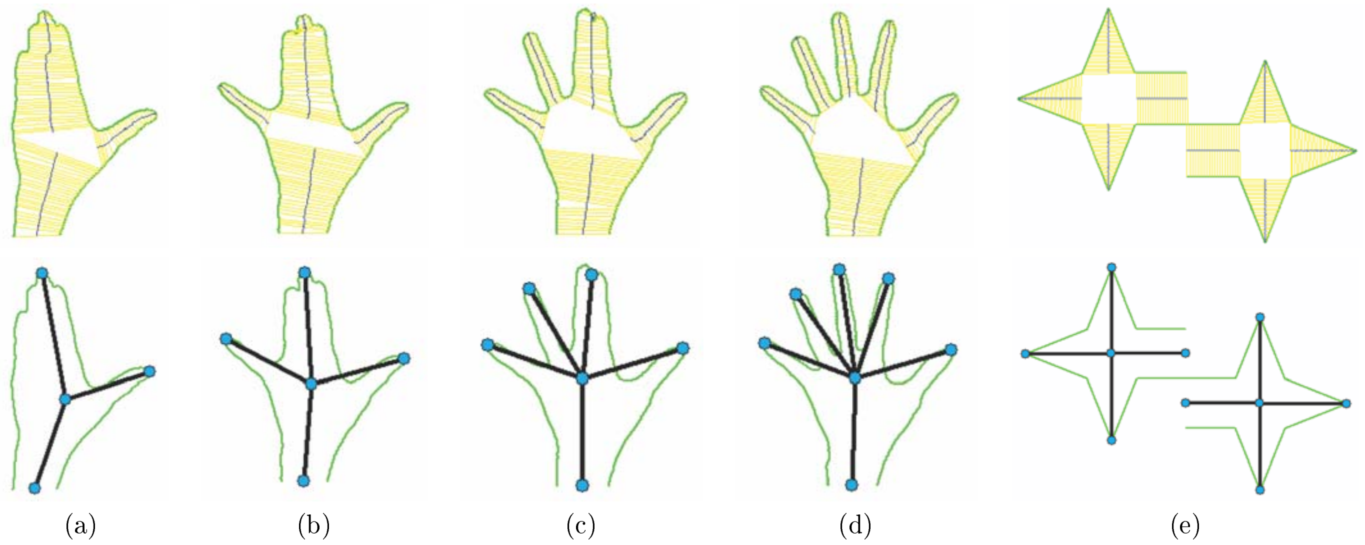


Fig. 11. (a)-(e)The algorithm can also be applied to open contours, but the SA-tree may be degenerated into an SA-forest as in (e).

Since the local and global information of a shape is encoded compactly using the SA-tree representation, comparing the similarity between two shapes can, therefore, be realized by measuring the degree of deformation of one shape into the other via matching their respective SA-trees. More precisely, we formulated an *approximate tree matching scheme* to compute the cost of matching one SA-tree to the other one. It is an exact method, that allow inexact matches (nodes in one tree do not have to match nodes in the other three) and gives the flexibility to matching two arbitrary SA-trees. In particular is applied when viewing position, stretching, articulation, and occlusion yield different shape contours, and thus different SA-trees even for the same object. There are many other approaches to shape matching according to various representation schemes of shapes. Closely related to the symmetry axis representation is the work by Siddiqi *et al.* [33] and the one by Pelillo, Siddiqi, and Zucker [24]. These approaches, though different, stress the importance of the symmetry axis representation.

With the SA-tree representation, we are able to utilize its free tree property and the topological structure information among neighboring nodes. Typical techniques such as pruning and merging vertices can be applied in the pro-

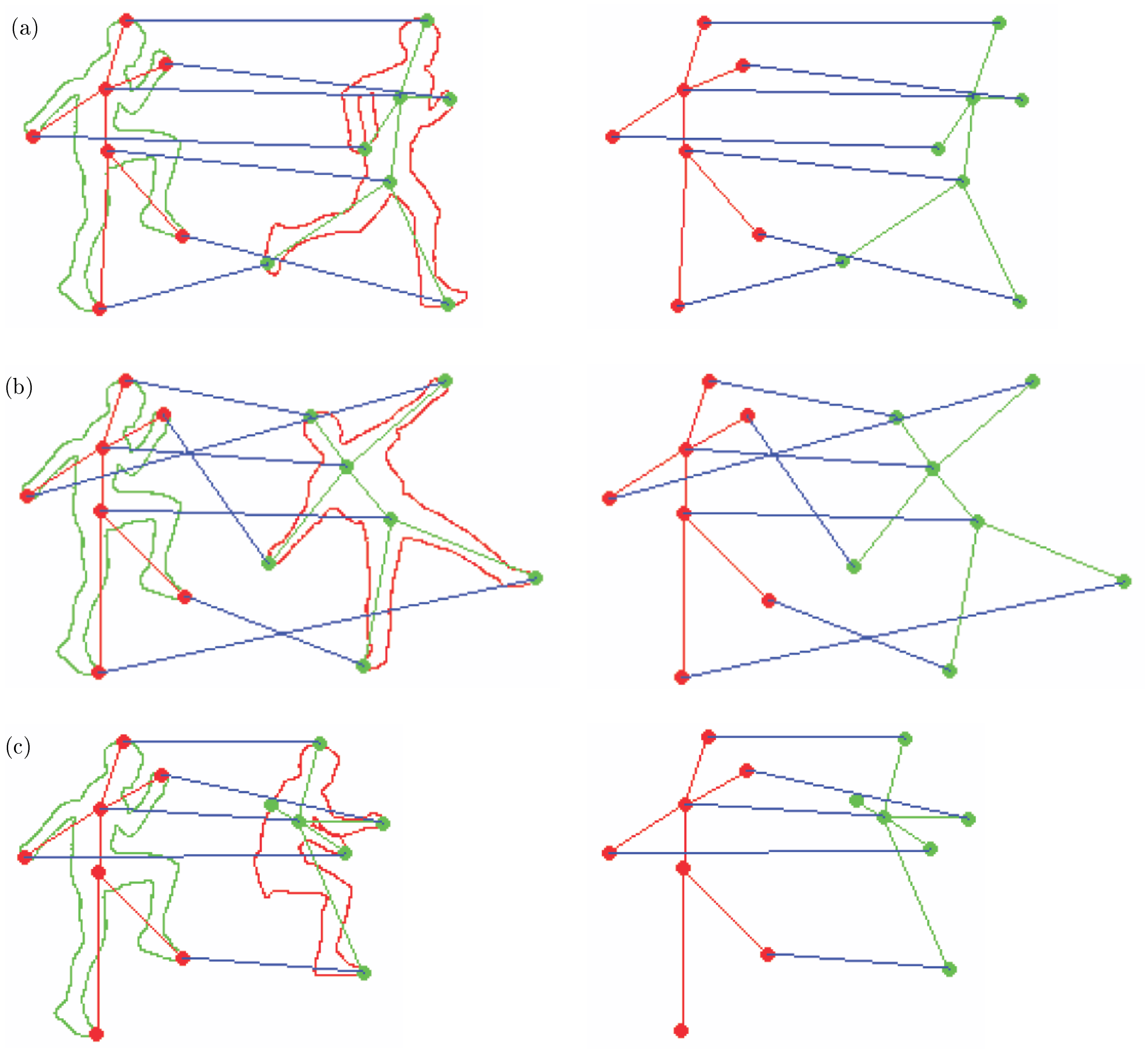
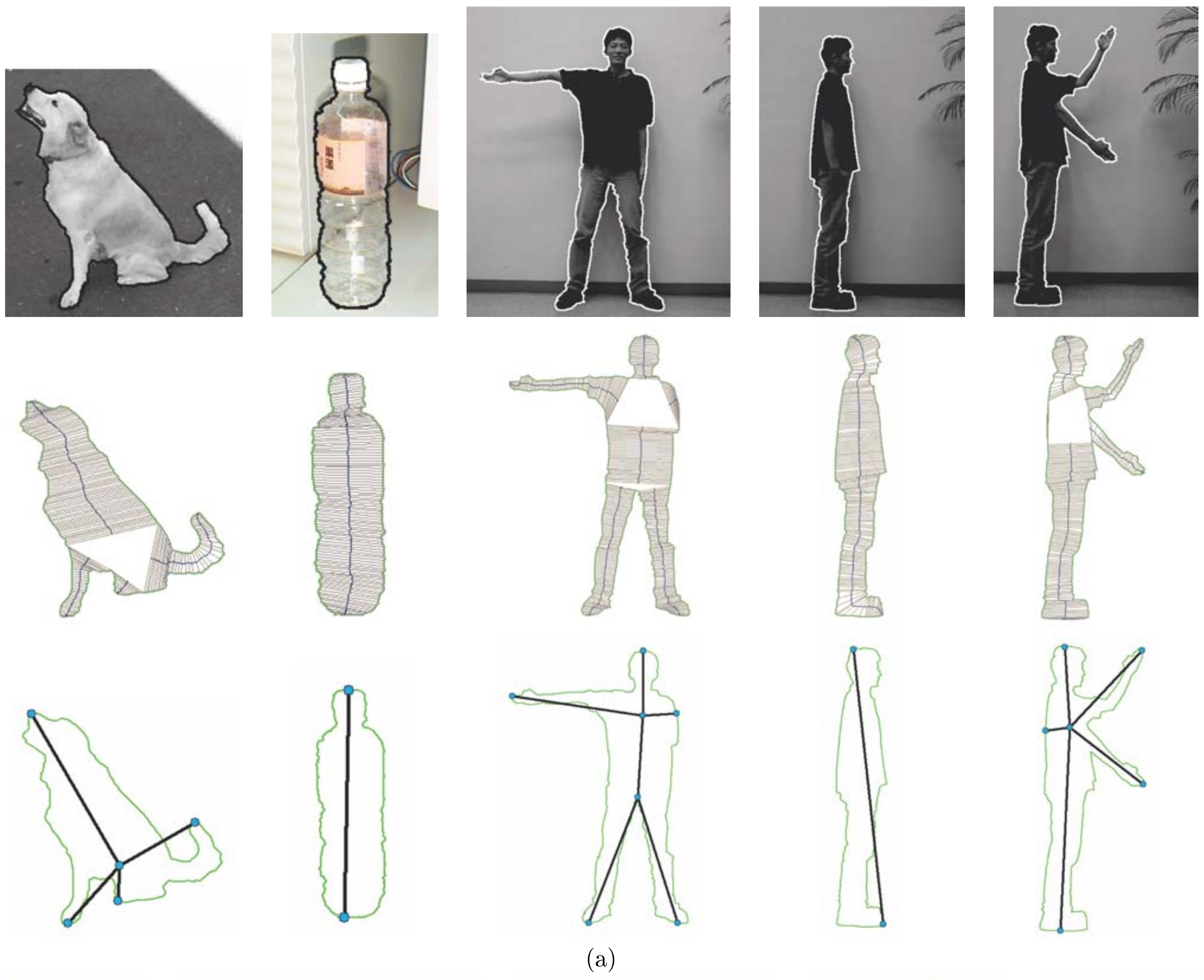


Fig. 12. The results illustrate that the SA-tree shape matching can account for articulation. Unlike typical string matching algorithms, the derived matching is not restricted to following a sequential ordering along the contours but considering a free tree structure. The difficult issue of occlusion can be handled appropriately by approximate tree matching with cut operation, as shown in (c). The degree of similarity decreases in the order of (a), (b) and (c) as their respective matching costs increase.

cess of matching to account for stretching and occlusion. Furthermore, since the actual geometric information and the best pairing pairs of points along a shape contour are encoded into the edges of an SA-tree, we can find not only the node-to-node but also the edge-to-edge/edge-to-path correspondences between two SA-trees. Such extension is more versatile than the regular approximate tree pattern matching [30]. The best approximate matching between two SA-trees can be found efficiently with an A^* algorithm. In Fig. 12, three matching results are shown to illustrate the matching scheme can generate good matching quality even when deformation, articulation and occlusion are present. In practice, to encode a collection of shapes into SA-trees can be done off-line. As to the SA-tree matching, it takes less than one minute to complete for shapes around the size of 1000 points, with a Pentium-III PC. Interested readers may find further details in [16].

V. Conclusion

We have proposed a new approach to shape representation. It is inspired by the idea of the symmetry axis [4]. The approach is the first one to be a variational one, i.e., the shape axis we obtain, given the shape, can be interpreted as the minimum solution to a well defined criterion. This is also where the power of the approach relies, where by changing the optimality criterion we can obtain different



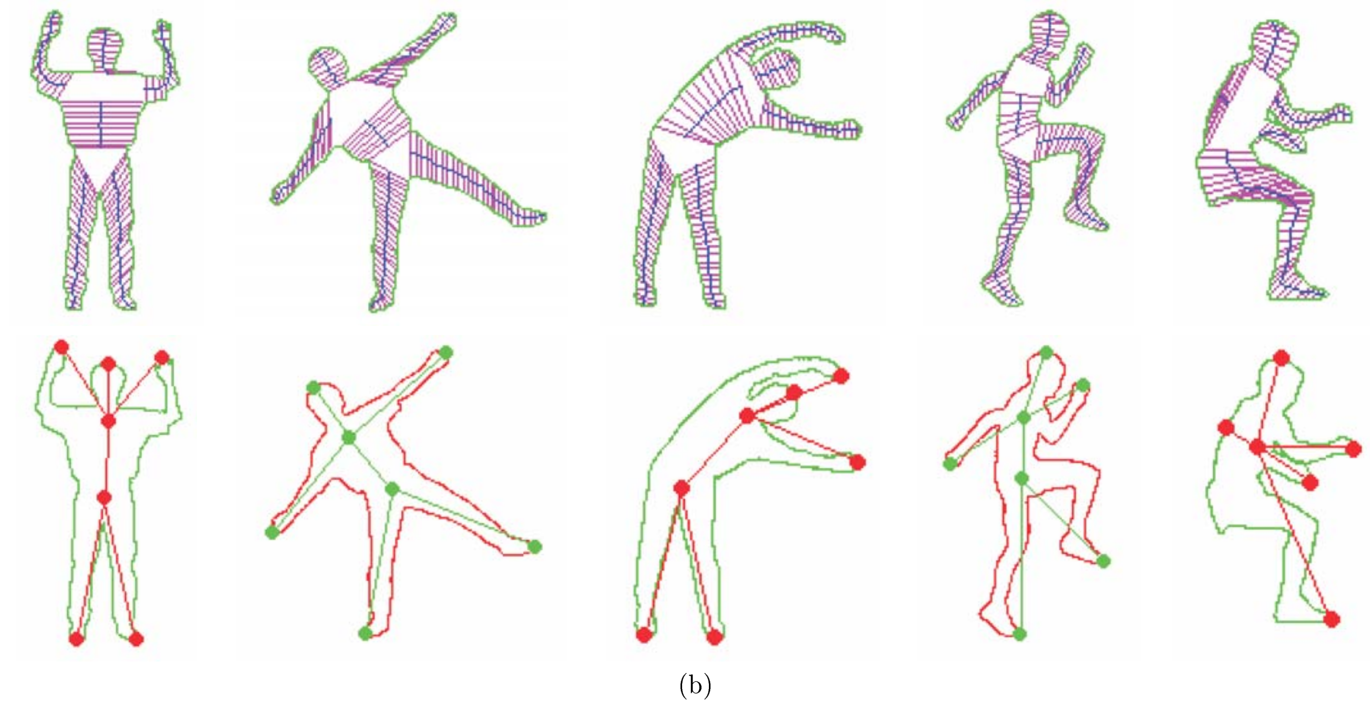


Fig. 13. (a) Shapes extracted from real images and their shape axes and SA-trees. (b) Examples of shape axes and SA-trees of various human shapes.

representations. Thus, one can (i) regularize the solution by adding penalties for bifurcations, (ii) add penalties for distances between the matched points (iii) vary the symmetry criterion, e.g., co-circularity, parallelism, add distance variation, add curvature terms. It is worth noticing that the number of branches in a bifurcation is allowed to be arbitrary, thus data driven.

The insight is to match a curve with itself, by establishing two different parameterizations (clockwise and counterclockwise) to the closed curve/shape and matching the two parameterizations. The optimal matching minimizes the variational criterion. The topological and geometrical constraints become matching constraints. We have given a precise formulation within the theory of matching curves, establishing the criteria/requirements on the form of the functional such that the solution is parameterization independent and scale independent.

We have also shown that a dynamic programming solution exists and it is guaranteed to find the optimal solution (shape axis). The complexity of the solution is $O(N^4)$ where N is the size of the discretized contour chain. Because the optimization criterion is written as a sum of positive costs, the graph where dynamic programming find the optimal solution have all positive value edges. This implies that the Dijkstra's algorithm can also be applied with more efficiency.

Overall we have described and demonstrated the efficiency and flexibility of our approach/algorithm on a variety of shapes, open or closed. An application to shape matching is also discussed to show the advantages of our proposed system over others.

Acknowledgments

We were grateful to conversations with Nava Rubin and the feedback from the reviewers. D.G. and T.L. were supported by NSF Career grant # 9733913 , DARPA and AFOSR grants F49620-96-1-0028 and F49620-96-1-0159, and R.V.K. by NSF DMS-0073047. T.L. was also supported in part by the Institute of Information Science, Academia Sinica of Taiwan and NSC grants 89-2213-E-001-023 and 89-2218-E-001-010.

References

[1] H. Asada and M. Brady, "The curvature primal sketch," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 2–14, January 1986.

[2] R. Basri, L. Costa, D. Geiger, and D.W. Jacobs, "Determining the similarity of deformable shapes," in *Physics Based Modeling Workshop in Computer Vision*, 1995.

[3] T.O. Binford, "Visual perception by computer," in *IEEE Conference on Systems and Control*, Miami, FL, 1971.

[4] H. Blum, "Biological shape and visual science," *Journal of Theoretical Biology*, vol. 38, pp. 205–287, 1973.

[5] C.A. Burbeck and S.M. Pizer, "Object representation by cores: Identifying and representing primitive spatial regions," *Vision Research*, vol. 35, 1995.

[6] T.H. Cormen, C.E. Leiserson, and R.L. Rivest, *Introduction to Algorithms*, McGraw-Hill, New York, 1990.

[7] Y. Gdalyahu and D. Weinshall, "Measures for silhouettes resemblance and representative silhouettes of curved objects," in *Proc. Fourth European Conf. Computer Vision*, Cambridge, England, 1996, vol. 2, pp. 363–375.

[8] R. Fawcett, A. Zisserman, and J.M. Brady, "Extracting structure from an affine view of a 3d point set with one or two bilateral symmetries," *Image and Vision Computing*, vol. 12, no. 9, pp. 615–622, November 1994.

[9] P.J. Giblin and G. Sapiro, "Affine invariant medial axis and skew symmetry," in *Proc. Sixth IEEE Int'l Conf. Computer Vision*, Bombay, India, 1998, pp. 833–838.

[10] E.C. Hildreth, *The Measurement of Visual Motion*, MIT Press, Cambridge, 1983.

[11] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge, "Comparing images using the hausdorff distance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 850–863, September 1993.

[12] D.P. Huttenlocher and S. Ullman, "Object recognition using alignment," in *Proc. First IEEE Int'l Conf. Computer Vision*, London, England, 1987, pp. 102–111.

[13] M. Kass, A.P. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int'l J. Computer Vision*, vol. 1, no. 4, pp. 321–331, January 1988.

[14] B.B. Kimia, A.R. Tannenbaum, and S.W. Zucker, "Shapes, shocks, and deformations i: The components of 2-dimensional shape and the reaction-diffusion space," *Int'l J. Computer Vision*, vol. 15, no. 3, pp. 189–224, July 1995.

[15] F. Leymarie and M.D. Levine, "Simulating the grassfire transform using an active contour model," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 1, pp. 56–75, January 1992.

[16] T-L. Liu and D. Geiger, "Approximate tree matching and shape similarity," in *Proc. Seventh IEEE Int'l Conf. Computer Vision*, Corfu, Greece, 1999, pp. 456–462.

[17] T-L. Liu, D. Geiger, and R. Kohn, "Representation and self-similarity of shapes," in *Proc. Sixth IEEE Int'l Conf. Computer Vision*, Bombay, India, 1998, pp. 1129–1135.

[18] R. Nevatia and T.O. Binford, "Description and recognition of curved objects," *Artificial Intelligence*, vol. 8, no. 1, pp. 77–98, 1977.

[19] R.L. Ogniewicz, *Discrete Voronoi Skeletons*, Hartung-Gorre, 1993.

[20] P. Parent and S.W. Zucker, "Trace inference, curvature consistency, and curve detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 8, pp. 823–839, August 1989.

[21] A.P. Pentland, "Recognition by parts," in *Proc. First IEEE Int'l Conf. Computer Vision*, London, England, 1987, pp. 612–620.

[22] A.P. Pentland and S. Sclaroff, "Closed-form solutions for physically based shape modeling and recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 715–729, July 1991.

[23] M. Pelillo, K. Siddiqi, and S.W. Zucker, "Matching hierarchical structures using association graphs," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 11, pp. 1105–1120, November 1999.

[24] M. Pelillo, K. Siddiqi, and S.W. Zucker, "Matching hierarchical structures using association graphs," in *Proc. Fifth European Conf. Computer Vision*, University of Freiburg, Germany, 1998.

[25] S.M. Pizer, W.R. Oliver, and S.H. Bloomberg, "Hierarchical shape description via the multiresolution symmetric axis transform," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 4, pp. 505–511, July 1987.

[26] J. Ponce and O.D. Faugeras, "An object centered hierarchial representation for 3d objects: The prism tree," *Computer Vision Graphics and Image Processing*, vol. 38, no. 1, pp. 1–28, April 1987.

[27] W.A. Richards and D.D. Hoffman, "Codon constraints on closed 2d shapes," *Computer Vision Graphics and Image Processing*, vol. 31, no. 3, pp. 265–281, September 1985.

[28] S. Sclaroff and A.P. Pentland, "Modal matching for correspondence and recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 6, pp. 545–561, June 1995.

[29] J. Segen, "Model learning and recognition of nonrigid objects," in *Proc. Conf. Computer Vision and Pattern Recognition*, San Diego, CA, 1989, pp. 597–602.

[30] D. Shasha, J. Wang, and K. Zhang, "Exact and approximate algorithm for unordered tree matching," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 24, no. 4, pp. 668–678, 1994.

[31] K. Siddiqi and B.B. Kimia, "Parts of visual form: Computational aspects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 3, pp. 239–251, March 1995.

[32] K. Siddiqi and B.B. Kimia, "A shock grammar for recognition,"

in *Proc. Conf. Computer Vision and Pattern Recognition*, San Francisco, CA, 1996, pp. 507–513.

[33] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S.W. Zucker, "Shock graphs and shape matching," in *Proc. Sixth IEEE Int'l Conf. Computer Vision*, Bombay, India, 1998, pp. 222–229.

[34] Z.S.G. Tari, J. Shah, and H. Pien, "Extraction of shape skeletons from grayscale images," *Computer Vision Image Understanding*, vol. 66, no. 2, pp. 133–146, May 1997.

[35] S. Tari and J. Shah, "Local symmetries of shapes in arbitrary dimension," in *Proc. Sixth IEEE Int'l Conf. Computer Vision*, Bombay, India, 1998, pp. 1123–1128.

[36] D. Terzopoulos, A.P. Witkin, and M. Kass, "Symmetry-seeking models and 3d object reconstruction," *Int'l J. Computer Vision*, vol. 1, no. 3, pp. 211–221, October 1987.

[37] S. Ullman, "Aligning pictorial descriptions: An approach to object recognition," *Cognition*, vol. 32, no. 3, pp. 193–254, 1989.

[38] F. Ulupinar and R. Nevatia, "Perception of 3-d surfaces from 2-d contours," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 1, pp. 3–18, January 1993.

[39] S.C. Zhu and A.L. Yuille, "Forms: A flexible object recognition and modelling system," in *Proc. Fifth IEEE Int'l Conf. Computer Vision*, Cambridge, MA, 1995, pp. 465–472.

Davi Geiger received the BS degree in physics from PUC-Rio, Brazil, in 1980, and the Ph.D. degree in Physics at MIT, in 1990, having developed his thesis at the Artificial Intelligence Laboratory at MIT. He is now an Associate Professor of Computer Science and Neural Science at New York University. He received the Career Award from National Science Foundation in 1998. The central theme of his research is the development of a theory of how the brain works and more generally how information is processed. The domain of interests include computer vision, learning theory, memory and its applications.

Tyng-Luh Liu received the BS degree with honors in applied mathematics from National Chengchi University, Taipei, Taiwan, in 1986. Form 1986 to 1988, he was serving an obligatory military service in the Air Force of Taiwan. He received the MS degree in Mathematics in 1990, MS degree and PhD degree in computer science in 1995 and 1997, all from New York University. After one year as a post-doctoral researcher at the Courant Institute of Mathematical Sciences, he joined the Institute of Information Science at Academia Sinica, Taiwan, in 1998, and now holds a tenure-tracked assistant research fellow position. His research interests include computer vision, learning theory, computer graphics, and more recently bioinformatics.

Robert V. Kohn is a Professor of Mathematics at New York University's Courant Institute of Mathematical Sciences. He received his A.B. summa cum laude from Harvard (1974), his M.Sc.from the Warwick (1975), and his Ph.D. from Princeton (1979), all in mathematics. He was awarded the Ralph Kleinman Prize by the Society for Industrial and Applied Mathematics in 1999; prior recognitions include an NSF Postdoctoral Fellowship (1979-81) and a Sloan Research Fellowship (1984-86). His research addresses nonlinear partial differential equations and the calculus of variations, focusing mainly on problems from finance, materials science, mechanics, physics, and vision.