IEEE Trans on PAMI, Vol. 21, No.11, November 1999.

Stochastic Jump-diffusion Process for Computing Medial Axes in Markov Random Fields

Song-Chun Zhu

Department of Computer and Information Sciences
Ohio State University
Columbus, OH 43210
szhu@cis.ohio-state.edu
http://www.cis.ohio-state.edu/~szhu/

Abstract

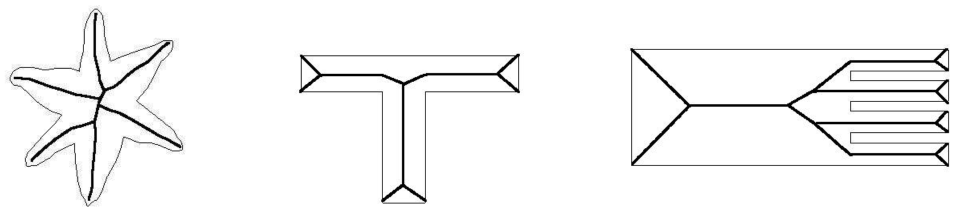
This paper proposes a statistical framework for computing medial axes of 2D shapes. In this paper the computation of medial axes is posed as a statistical inference problem not as a mathematical transform. This paper contributes to three aspects in computing medial axes. I) Prior knowledge is adopted for axes and junctions so that axes around junctions are regularized. II) Multiple interpretations of axes are possible, each being assigned a probability. III) A novel stochastic jump-diffusion process is proposed for estimating both axes and junctions in Markov random fields. We argue that the stochastic algorithm for computing medial axes is compatible with existing algorithms for image segmentation, such as region growing[31], snake[7] and region competition[26]. Thus out method provides a new direction for computing medial axes from texture images. Experiments are demonstrated on both synthetic and real 2D shapes. This algorithm has been successfully applied to shape learning and sampling in a companion paper[30].

Keywords: Medial axis transform, jump-diffusion process, energy minimization, Markov random field.

1 Introduction and motivations

Medial axis or symmetry axis transform is an important representation of object shapes[13], and it has been extensively studied in both human and computer vision following the pioneering work of Blum etc.[1, 19]. Existing algorithms for computing medial axes are divided into four categories: 1). active contour method[12], 2). Voronoi diagram[20], 3). shape evolution by partial differential equations and level sets[9, 21], and 4). tracking deformable circles[27]. Some algorithms have been proposed for computing segments of symmetry axes from grey level images, using difference of low-pass filters[3], annular symmetry operators following edge detection[8], and diffusion equations[22]. Medial axes have also been successfully used in object recognition, for example, the FORMS system by Zhu and Yuille transforms 2D shapes into skeleton graphs, and objects are recognized in a bottom-up/top-down loop[27]. Recently Liu, Geiger and Kohn have studied perhaps the first variational method for extracting the skeleton tree for object recognition[16].

As medial axis is notoriously sensitive to boundary noise, the main thrust of existing algorithms for extracting medial axis is to overcome the robustness problem, and various methods have been proposed, such as trimming extra branches[20], smoothing boundaries by diffusion equations[9, 23] or using deformable circles instead of hard disks[27]. The significance of an axis branch can also be measured[12]. Though this problem has been well addressed, there are some other problems which have received less attention in the literature.



a b c

Figure 1 Medial axes defined on maximum disks.

Firstly, object shapes often have *multiple interpretations* in terms of medial axes, and ambiguities are especially pronounced around corners and junctions. For example, figure 1.a displays the medial axis of a flower using the maximum disk definition. Semantically it seems desirable that all six parts of the flower join at one point and that the angles between adjacent axes be approximately equal (see figure 15). For more examples, the medial axes for the T-shape and the hand-shape are displayed in figure 1.b and figure 1.c respectively. Other possible interpretations of the two shapes in terms of axes are shown in figure 14.

The perception of medial axes largely depends on the basic shape elements used in object recognition. Thus in a *bottom-up* process, it is best to define medial axes in a probabilistic language, so that multiple interpretations are preserved for decisions by the high level recognition process. In contrast, the current deterministic definition of medial axes lacks such flexibilities. Furthermore the medial axis transform is unstable at junctions of degree larger than three, because it is rare to have four or more tangent points on the shape sharing one maximum circle. Therefore some regularizations are necessary.

Secondly, it is unclear how medial axes can be reliably computed from texture images. We argue that the computation of medial axes should interact closely with the image segmentation process.

This argument is supported by both psychological and neurophysiological experiments[10, 11]. In psychology, Kovacs and Julesz studied the contrast-sensitivity thresholds to the perception of Gabor filter patches[10] in pre-attentive human visual perception. They arranged some Gabor filter patches to form a closed contour among other randomly placed similar Gabor patches. They found that the sensitivity at the points of medial axes of closed contours was remarkably increased, and the task was performed in the pre-attentive vision stage. In neurophysiology, Lee et. al. discovered that the responses of some neurons in the primary visual cortex (V1) of a monkey were particularly accentuated at medial axes of textured shapes in the very early stage (80-200 milliseconds) [11].

From a computational point of view, medial axes provide region-based grouping information – symmetry, parallelism, and proximity, which are important for obtaining accurate segmentation. Thus it is beneficial for image segmentation algorithms if they interact with the on-line computation of medial axes. However, there is a major gap between existing algorithms for computing medial axes and algorithms for image segmentation. The former are formulated in a deterministic setting, whereas the latter are mostly studied in a statistical framework, for example, region growing[31], active contour (SNAKE)[7], and region competition[26]. This incompatibility poses a major barrier for extracting medial axes and object shapes from texture images.

Motivated by the above observations, we propose to define medial axes on Markov random fields, and to pose the computation of medial axes as a statistical inference problem. Computing medial axes in a statistical framework has the following potential advantages.

1. Prior probability models are introduced to regularize ambiguities of axes and junctions. These prior models are characteristic of generic properties shared by natural object shapes. For example, prior models (or energy terms) can be added to favor axis smoothness, symmetry and alignment at junctions, such as "T", "X" and "Y" junctions.

2. Stochastic algorithms, such as the Gibbs sampler[4] and the Metropolis algorithm[17], could be utilized for inference. As these stochastic algorithms become more and more important in achieving high quality image segmentation, a stochastic algorithm for computing medial axes could be integrated into the processes of image segmentation and perceptual organization.

3. The statistical formulation makes it possible to compute multiple interpretations of a shape in terms of medial axes, each being assigned a probability.

In the computer vision literature, we have noticed some interesting work along this line. In 1984, Tsao and Fu published perhaps the first paper on skeleton of stochastic nature[24]. In [24], a skeleton graph is computed in a *deterministic* way based on maximum disks, and then random shape instances are generated by either perturbing the coordinates of the points on the skeleton or changing the radius at a skeleton point. In 1992, Li proposed an energy minimization formulation for matching two curves represented as graphs in studying object recognition from range data[14, 15]. Most recently Liu, Geiger and Kohn[16] proposed an interesting energy minimization method for matching a curve to itself. They successfully adopted a dynamic programming method, and their energy function is similar in spirit to the one used for computing the mapping function in this paper. Their work was published independent of the conference version of this paper[29].

In this paper, we propose a stochastic algorithm for computing medial axes in two phases. 1). In phase I, a Metropolis-Hastings algorithm[17] is adopted to compute a mapping function which maps each linelet in the 2D shape to $n \geq 0$ other linelets in favor of parallelism and symmetry. II). In phase II, a stochastic jump-diffusion process is designed to compute axes and junctions. This jump-diffusion process simulates a Monte Carlo Markov chain traveling in a heterogeneous space of the medial axis representation.

The paper is organized as follows. Section (2) defines a piecewise continuous mapping function. Section (3) studies a stochastic algorithm for computing the mapping function, and section (4) discusses how to compute medial axis and junctions separately based on the mapping function. Section (5) studies a jump-diffusion process for computing the mapping function and junctions simultaneously. We demonstrate the performance of the algorithm on a variety of shapes in section (6). Section (7) concludes the paper with a discussion.

2 Define a mapping function on 2D shape

Let $\Gamma(s) = (x(s), y(s))$ denote a simple, closed 2D shape, where $s \in [0, 1]$ is the arc-length and $\Gamma(0) = \Gamma(1)$. We start with defining a symmetry mapping function,

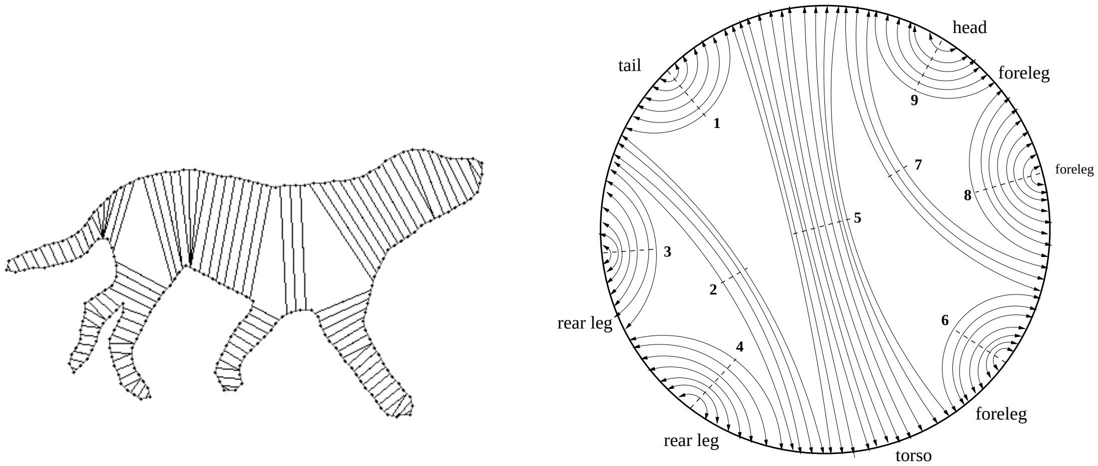
$$\phi(s) : [0, 1] \rightarrow [0, 1] \cup \{\text{nil}\}, \qquad \phi(\text{s}) = \text{t} \Longleftrightarrow \phi(\text{t}) = \text{s}, \quad \text{if s, t} \neq \text{nil}.$$

$\phi(s)$ divides the circular domain $[0, 1]$ into a set of disjoint intervals $\mathcal{I} = \{(s_{i0}, s_{i1}) \subset [0, 1]\}$, and establishes a piecewise continuous mapping between them. If $\phi(s) = t \neq$

nil, the linelets at points s and t, denoted by Γ_s and Γ_t respectively, are said to be matched[1]. If $\phi(s) = \text{nil}$, it means Γ_s has no good matches and it remains as a single linelet. Usually $\phi(s) = \text{nil}$ for point s around a junction or near the end of a shape branch.

For example, a dog shape is shown in figure 2.a with a mapping function $\phi(s)$ displayed by the line segments, and more examples are displayed in figure 4 and figure 5. In these figures, we notice that some linelets are not matched and that some have multiple matches due to discretization. Figure 2.b is an abstract representation of the mapping function $\phi(s)$.

Definition: *An interval* $(s_0, s_1) \in \mathcal{I}$ *is called an* empty interval*, if* $\phi(s) = \text{nil}, \forall \text{s} \in (\text{s}_0, \text{s}_1)$*, and other intervals in* $\mathcal{I}$ *are called* matched intervals*. We denote by* $\mathcal{B}_{me}$ *the set of breaking points between adjacent matched intervals and empty intervals, and* $\mathcal{B}_{mm}$ *the set of breaking points between adjacent matched intervals.*



a b

Figure 2 An example, a). a mapping function $\phi(s)$ on a dog shape, b), the abstract representation of $\phi(s)$ as an adjacency graph with 18 matched intervals, 0 empty intervals, and 18 break points in $\mathcal{B}_{mm}$.

[1] As the mapping function $\phi(s)$ is piecewise continuous, the linelet Γ_s is of infinitesimal length. Γ_s denotes $(x(s), y(s))$ and the tangent direction at s. In implementation, the curve is divided into N linelets of equal length.

Intuitively the mapping function $\phi(s)$ should satisfy the following properties.

1. If $\phi(s) = t \neq \text{nil}$, then the line segment connecting linelets Γ_s and Γ_t should not intersect any other mappings and it should contain no points outside the shape $\Gamma(s)$. This property favors selecting a mid-chord symmetry axis[1] instead of centers of maximum disks or other definitions[25].

2. If $\phi(s) = t \neq \text{nil}$, then the linelets Γ_s and Γ_t should be close in space and be approximately symmetric and parallel to each other in orientations.

3. If $\phi(s)$ is a continuous mapping between intervals $(s_1, s_2) \Longleftrightarrow (t_1, t_2)$, then $\phi(s)$ should be as uniform as possible. That is, $\frac{d\phi}{ds}$ should be close to -1.

4. The number of breaking points in $\phi(s)$ should be as small as possible.

The first property is a hard constraint and we define the set of all valid mapping functions for a shape Γ as,

$$\Omega_\phi = \{\phi(s) : \phi(s) \text{ satistifies property } 1\}. \tag{1}$$

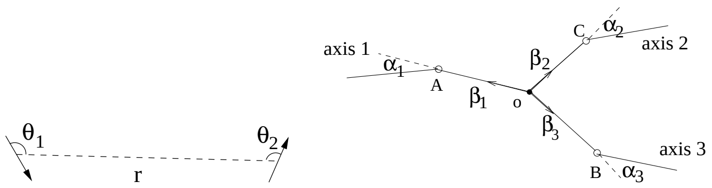
We define a probability distribution on Ω_ϕ so that properties 2, 3, and 4 are observed,

$$p(\phi|\Gamma) = \frac{1}{Z} \exp^{-E(\phi|\Gamma)}, \quad \forall \phi \in \Omega_\phi. \tag{2}$$

The energy functional is chosen to be

$$E_1(\phi|\Gamma) = \oint_{\phi(s)\neq \text{nil}} C(s, \phi(s)) + \alpha ||\nabla\phi(s) + 1|| ds + \gamma \oint_{\phi(s)=\text{nil}} ds + \gamma_{me}||\mathcal{B}_{me}|| + \gamma_{mm}||\mathcal{B}_{mm}||. \tag{3}$$

In the above energy functional, $\gamma_{me} > 0$ and $\gamma_{mm} > 0$ penalize short intervals and force long axes to form as suggested by property 4. γds is the cost for an unmatched linelet. If $\gamma = 0$, then all linelets remain singles, and as γ increases, more and more linelets are matched. The effect of γ is illustrated in figure 5. Inside a matched interval, α favors uniform mappings following property 3, and $C(s, \phi(s))ds$ is the cost for mapping two linelets at s and $\phi(s)$. Energy functionals of similar nature have been previously adopted in image segmentation and restoration[4, 18].



a b

Figure 3 a). the mapping between two linelets, b). a junction of degree=3.

Now we define $C(s, \phi(s))$. Figure 3.a shows two linelets linked by a dashed line, whose length is normalized by L – the length of $\Gamma(s)$:

$$r(s) = \|\Gamma(s) - \Gamma(\phi(s))\|/L.$$

So $r(s)$ is invariant to 2D similar transforms (translation, rotation, and scaling). The two linelets have tangent angles θ_1, θ_2 relative to the dashed line. We denote by θ_p and θ_s the continuous measures for parallelism and symmetry between the two linelets respectively, $\theta_p = \theta_1 + \theta_2 - \pi, \quad \text{and} \quad \theta_s = \theta_1 - \theta_2$.

Following property 2 of $\phi(s)$, we choose

$$C(s, \phi(s)) = \frac{\theta_p^2}{2\sigma_p^2} + \frac{\theta_s^2}{2\sigma_s^2} + \mu r(s). \tag{4}$$

The parameters are chosen empirically, and we fix $\sigma_p = \pi/10$, $\sigma_s = \pi/8$ and $\mu = 100$ in all our experiments in this paper. A rigorous account for the sensitivity of the algorithm to the selection of these parameters has yet to be studied. In theory, the precise function for $C(s, \phi(s))$ can be learned from training examples in a non-parametric form based on the maximum entropy principle[30].

Remark I. The probability $p(\phi|\Gamma)$ and the energy $E(\phi)$ is invariant to translation, rotation, and scaling.

Remark II. $\Gamma(s)$ is a one-dimensional random field with $(x(s), y(s))$ being the random variables. Thus $\phi(s)$ *specifies an adjacency graph – or a neighborhood structure on this random field*, illustrated in figure 2.b. In this graph, spatially adjacent points become near neighbors, although they may be far away from each other along the contour. So region based information, such as parallelism and symmetry, is characterized. A probabilistic shape model defined on such graphs is studied in a companion paper[30].

3 Compute the mapping function

To compute the mapping function $\phi(s)$, we discretize a shape $\Gamma(s)$ into $N = L/c$ linelets, $\Gamma_0, \Gamma_1, ..., \Gamma_{N-1}$, of equal length. The constant c should be chosen to be small for good resolution, but the tangents of linelets cannot be computed reliably if c is too small. We choose $c = 6$ pixels in this paper. A problem with this equal length discretization is that many-to-one mappings become necessary. These many-to-one mappings often occur when the curvatures of two matched intervals are very different. Now the mapping function $\phi(s)$ is discretized into a mapping graph (see in figure 2.a and figure 4) which is defined below.

Definition: *Let* $G =< V, E \cup M >$ *denote a* mapping graph *for shape* Γ*, with* $V = \{\Gamma_0, ..., \Gamma_{N-1}\}$ *being the set of vertices for the linelets,* $E = \{< \Gamma_i, \Gamma_{(i+1) mod\, N} >, \; i = 0, 1, ..., N-1\}$ *the set of edges connecting the linelets along the contour, and* $M = \{< \Gamma_i, \Gamma_j >\}$ *the set of edges for the mapping function* $\phi(s)$*.* M *is said to be valid if* G *is a planar graph and all edges in* M *lie inside the shape* Γ*. The* mapping degree *of a linelet* Γ_i*, denoted by* $d(\Gamma_i)$*, is the number of edges in* M *that end at* Γ_i*.*

The energy functional $E_1(\phi|\Gamma)$ in equation (3) is discretized as,

$$E_1(M|\Gamma) = \{ \sum_{<\Gamma_i,\Gamma_j>\in M} C(\Gamma_i, \Gamma_j)/d(\Gamma_i) + \alpha \sum_{d(\Gamma_i)\neq 0} \|d(\Gamma_i) - 1\| + \gamma \|U\| \} ds + \gamma_{me}\|B_{me}\| + \gamma_{mm}\|B_{mm}\|. \tag{5}$$

where U is the set of unmatched linelets, $ds/d(\Gamma_i)$ is the effective arc length for mapping two linelets Γ_i, Γ_j. $ds = 1/N$ is the unit length of a linelet. If Γ_i is matched $d(\Gamma_i)$ times, then Γ_i is divided into $d(\Gamma_i)$ virtual linelets each of which has arc length $ds/d(\Gamma_i)$.

For any two sequential linelets Γ_i, Γ_{i+1}, we say that a mapping M is *continuous* between Γ_i and Γ_{i+1} if either Γ_i and Γ_j are matched to nil or Γ_i and Γ_{i+1} are matched to the same linelet or two sequential linelets.

The matching cost between any two linelets Γ_i and Γ_j is,

$$C(\Gamma_i, \Gamma_j) = \begin{cases} \infty & \text{if } <\Gamma_i, \Gamma_j> \text{ is an invalid match, otherwise,} \\ C(i,j) & \text{(see equation (4)).} \end{cases}$$

The number of possible matches for each linelet could be as large as N (including nil), but in practice we only need to consider the $m \ll N$ most promising candidates.

Definition. *For each linelet* Γ_i*, a set* S_i *of* matching candidates *includes: 1. the null match* nil*, 2.* $j \in S_i$ *if the edge* $<\Gamma_i, \Gamma_j>$ *is inside the shape* $\Gamma(s)$*.*

Therefore the optimal mapping M^* is computed by maximizing the probability $p(M|\Gamma) = \frac{1}{Z}e^{-E_1(M|\Gamma)}$ in the following space,

$$\Omega_M = \{\ M \in \prod_{i=0}^{N-1} \{(\Gamma_i, \Gamma_j) : j \in S_i\}\ :\ M \text{ is valid }\},$$

$$M^* = \arg\max_{M \in \Omega_M} p(M|\Gamma) = \arg\min_{M \in \Omega_M} E_1(M|\Gamma).$$

Note that $M, \Omega_M, p(M|\Gamma)$, and $E_1(M|\Gamma)$ are the discrete descriptions of $\phi(s), \Omega_\phi, p(\phi|\Gamma)$, and $E_1(\phi|\Gamma)$ respectively.

Due to hard constraints and many-to-one mappings, dynamic programming is not applicable. We adopt a Metropolis-Hastings algorithm for optimal solution[17]. The algorithm starts with an M where all linelets being matched to nil, and it flips one edge $e =$ on and $e =$ off at each step. This algorithm can also be viewed as a Gibbs sampler[4].

Algorithm I: A Stochastic Algorithm for the Mapping Function Φ

0\. Partition Γ into $N = L/c$ linelets $\Gamma_i, i = 0, 1, ..., N-1$.
1\. for each Γ_i, $i = 0, 1, ..., N-1$, Compute the candidate set S_i.
2\. $M \leftarrow \emptyset$, *count* $\leftarrow 0$.
3\. Pick up $i \in \{0, 1, ..., N-1\}$ with a uniform distribution.
4\. Pick up $j \in S_i$ according to $q_i(j)$, i.e. $e =< \Gamma_i, \Gamma_j >$.
5   if e intersects existing mappings, go to 10.
6   Compute $p(e = \text{on}|M)$ and $p(e = \text{off}|M)$.
7   Draw a random variable $r \in [0, 1]$ at uniform distribution
8\.   If $r < p(e = \text{on}|M)$, then $M \leftarrow M \cup \{e\}$
9\.   otherwise $\qquad\qquad\quad M \leftarrow M/\{e\}$.
10\.   *count* $\leftarrow$ *count* $+ 1$.
11\. If *count* $<$ *threshold* Goto 3, else stop.

In step 5, the Markov chain is confined by the hard constraint. In step 4, the algorithm proposes Γ_j according to a probability $q_i(j)$ for $j \in S_i$. The efficiency of the algorithm can be improved if promising candidates are nominated more frequently. For example, $q_i(j) = \frac{1}{Z_i} \exp^{-C(\Gamma_i, \Gamma_j)}$, with $Z_i = \sum_{j \in S_i} \exp^{-C(\Gamma_i, \Gamma_j)}$ being a normalizing constant. In practice, we found that the probability mass of $q_i(j)$ above is almost completely focused on the first m candidates. Thus we set $q_i(j)$ to be a uniform distribution on a subset $\hat{S}_j = \{\mathrm{nil}, \mathrm{i_1}, \mathrm{i_2}, ..., \mathrm{i_m}\} \subset \mathrm{S_j}$ as a trade-off between efficiency and simplicity. The size m is selected according to the "cutoff" probability in $q_i(j)$. In our experiments, thanks to the smoothness of the shape, $m = 5$ is found to be enough.

In the above algorithm, we define flipping N edges as one sweep. An annealing schedule is introduced to achieve optimal solution, and we set the temperature $T = 15e^{-sweep/300}$, which is not crucial for most shapes. The computational complexity is in the order of the number of sweeps multiplied by N. The algorithm stops after a conservative threshold of 1000 sweeps. As N is on the order of 200 to 400, the computation is very fast–e.g. a few seconds in a SGI O2, although annealing is used.

We show some mapping results in figure 2.a, figure 4, and figure 5. In figure 5, the *most probable mapping* changes from a mid-chord symmetry axis to the centers of maximum circles as γ increases. This shows an interesting difference between a geometric definition of medial axis and a statistical definition.

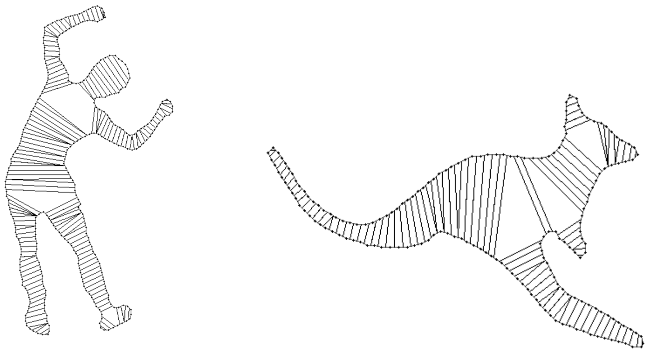
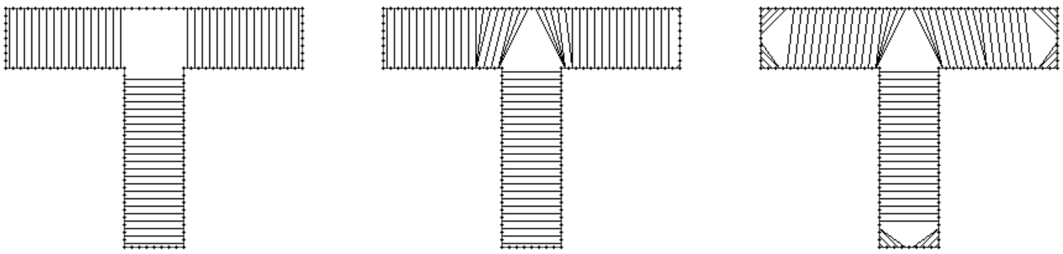


Figure 4 Two examples of the computed mapping graph G.



a b c

Figure 5 As γ is the penalty for a linelet not being matched, increasing γ will force all linelets to be matched, a). $\gamma = 12$, b). $\gamma = 22$, c). $\gamma = 27$.

In the rest of this section, we discuss how $p(e = \text{on/off} \,|\, M)$ is computed and analyze the Markov property of $p(M|\Gamma)$, that is, in $p(M|\Gamma)$ the state of an edge $e =< \Gamma_i, \Gamma_j > \in \{\text{on, off}\}$ only depends on its local neighborhood graph $\mathcal{N}_e$. Unlike the fixed neighborhoods in classical Markov random field models[4], $\mathcal{N}_e$ for each mapping edge changes dynamically.

Definition. *Let* $G =< V, E \cup M >$ *be a valid mapping graph,* $e =< \Gamma_i, \Gamma_j >$ *a mapping edge, and* $G^e =< V, E \cup M^e >$ *a subgraph where all mapping edges ending at* Γ_i *and* Γ_j *are removed. A polygon* $P^e =< v_0, v_1, ..., v_k, v_0 >$ *is defined as a closed path in* G^e *which passes vertices including* Γ_i *and* Γ_j*. A* minimum polygon P^e_{min} *is the polygon* P^e *with the shortest length.*

For example, figure 6 shows part of a subgraph $G^e =< V, E \cup M^e >$ for $e =< X, Y >$. The minimum polygon for e is $P^e_{min} =< C, X, D, H, I, Y, J, K, C >$. As G is a planar graph, it is easy to see that there is only one P^e_{min} for each e.

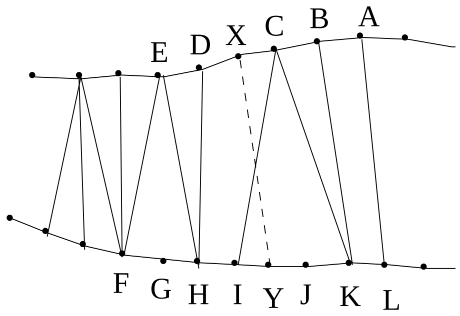
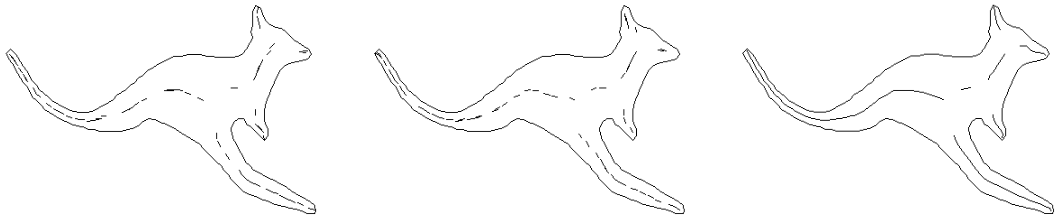


Figure 6 The definition of minimum polygon for a mapping edge $< X, Y >$.

Definition *Given* $G =< V, E \cup M >$ *and* e*, the neighborhood* $\mathcal{N}_e$ *is the minimum subgraph that contains all vertices in* P^e_{min} *and edges connecting two vertices in* P^e_{min}.

To judge whether e is a valid mapping edge or not, one only needs to see if e intersects mapping edges in $\mathcal{N}_e$. Thus it is easy to prove that $p(e|M) = p(e|\mathcal{N}_e)$. When $M = \emptyset$, $\mathcal{N}_e$ is the whole graph, as $|M|$ grows, the size of the neighborhood $\mathcal{N}_e$ for each edge e decreases rapidly. Therefore, under the hard constraints, $p(M|\Gamma)$ is a Markov random field model with neighborhood changing dynamically.

4 Computing axes and junctions



a b c

Figure 7 The symmetry axes of a kangaroo shape at time steps: a). 5 sweeps, b). 10 sweeps and c). 1000 sweeps.

In a mapping function $\phi(s)$, each pair of matched intervals correspond to a symmetry axis of the shape. In the mapping graph G, these axes emerge automatically if we simply connect the centers of the mapping edges $e \in M$ if M is continuous at e. Figure 7 displays the axes of a kangaroo shape at 5, 10, 1000 sweeps respectively. At the beginning, there are many scattered short axes. Later long axes emerge since they are favored in the energy function.

In general we should allow the points on the axes to slide along the mapping edge $e \in M$ for smoothness. This could be done by introducing a prior model based on the curvature of the axes. However in our experiments, the axes computed above are very smooth already – perhaps because the shape boundary is smooth. It is not necessary to adjust points on the axes.

Once we have computed the mapping graph $G =< V, E \cup M >$ and the axes, the next step is to compute junctions connecting adjacent axes. The junctions can be easily detected from the planar graph G. For example, figure 2.b shows a mapping graph, a portion of which is shown in figure 8. We first find the set $\mathcal{B}$ of break points in the mapping graph, e.g. $\mathcal{B} = \{A, B, ..., L\}$ in figure 8.

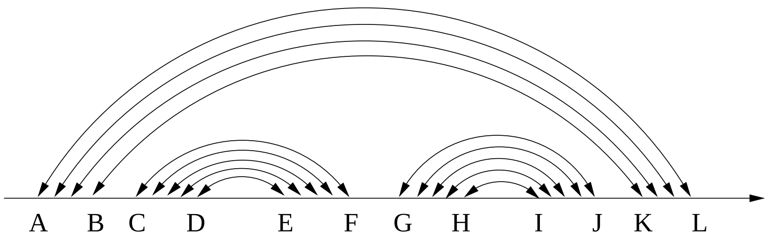


Figure 8 Detecting junctions.

We define a partition

$$\mathcal{B} = \cup_{i=1}^{K} \mathcal{B}_i,$$

and each $\mathcal{B}_i$ is a minimum polygon (or shortest loop) in the mapping graph G which pass only break points. For example, there are three minimum polygons in figure 8, $\mathcal{B}_1 =< B, K, J, G, F, C, B >$, $\mathcal{B}_2 =< D, E, D >$, $\mathcal{B}_3 =< H, I, H >$. $\mathcal{B}_2$ and $\mathcal{B}_3$ define two junctions of degree=1 each of which corresponds to the end of a shape branch. $\mathcal{B}_1$ is a junction of degree=3.

Given Γ and a mapping function $\phi(s)$ or M, the number of junctions K is determined, and the position (x_i, y_i) $i = 1, 2..., K$ of each junction is confined to a domain D_i bounded by the minimum polygon P^e_{min} as well as Γ. For example, the shadowed area in figure 9 shows D_i for a junction degree of 3.

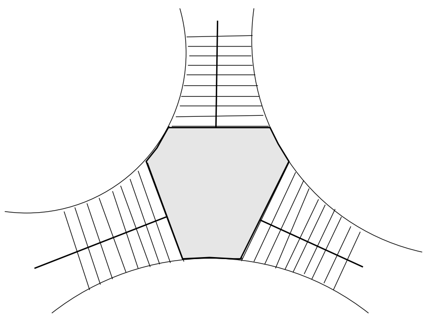


Figure 9 The shadowed area is the domain D_i for the position of a junction (x_i, y_i).

A conditional probability is defined for the set of junctions $J = \{(x_i, y_i), i =$

$1, 2, ..., K\}$,

$$p(\{x_i, y_i\}_{i=1}^K \,|\, M, \Gamma) \quad \text{on} \quad \Omega(J) = D_1 \times D_2 \times \cdots \times D_K.$$

In this section, we only discuss the computation of a single junction, and we shall discuss how to compute all the random variables $M, K, \{(x_i, y_i), i = 1, 2, ..., K\}$ in one run in the next section.[2]

As we discussed in section (1), medial axis transforms are often ill-defined around junctions, and this problem can be released by introducing *a priori* knowledge for axes near junctions. We may define a few categories of junctions such as "L", "T", "Y", "X" junctions, and find (x, y) that best fits to one of these models. In this paper, we choose a different method that computes junctions from a general energy function.

Figure 3.b displays a junction O connecting three axes at end points A, B, and C. The line segments OA, OB, and OC are oriented at angles $\beta_1, \beta_2, \beta_3$ respectively, and the angles between OA, OB, OC and the three axes are respectively $\alpha_1, \alpha_2, \alpha_3$. We adopt three generic rules in regularizing the junction position (x, y).

1. It makes smooth connections to the axes, e.g. $\alpha_1, \alpha_2, ..., \alpha_d$ should be as small as possible.

2. Some of the axes should join at (x, y) symmetrically, e.g. $\Delta\beta_i = |\beta_{i+1} - 2\beta_i + \beta_{i-1}|$ should be close to zero for some i.

3. Some of the axes should be aligned with other axes, e.g. $|\beta_i - \beta_j| = \pi$ for some choice of $< i, j >$.

The above criteria are expressed in a probability distribution $p(x_i, y_i|M) =$

[2] For a junction of degree=1, we simply assign it as the end point of the symmetry axis.

$\frac{1}{Z_i} e^{-E_2(x_i, y_i)}$ for $(x_i, y_i) \in D_i$, where the energy is,

$$E_2(x, y) = \sum_{i=1}^{d} a|\alpha_i|^2 + \sum_{i=1}^{d} b\psi(\Delta\beta_i) + \sum_{<i \neq j>} c\psi(|\beta_i - \beta_j| - \pi). \tag{6}$$

$\psi(\xi) = 1 - \frac{1}{1+(\xi/\delta)^\gamma}$ is a T-shaped function shown in figure 10. It has flat tails and a steep cusp around zero, so it forces ξ to zero if $|\xi|$ is very small, otherwise it has very small effect on ξ. The form of this function was suggested by the generic prior model learned from natural images[28].

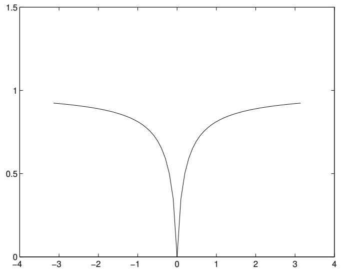


Figure 10 A T-function $\psi(\xi) = 1 - \frac{1}{1+(\xi/\delta)^\gamma}$.

The optimal (x, y) is computed by minimizing $E_2(x, y)$ for each junction separately. To achieve a global minimum, we adopt a stochastic algorithm which flips point (x, y) in a lattice using a Gibbs sampler[4]. In our experiments, we simply set $b = 0$, i.e. the second term is eliminated without observable difference.

Algorithm II: A Stochastic Process for Computing Junctions

0. Given Γ, M, find $K(M)$ junctions in the planar graph G.
1. For $i = 1, 2, ..., K(M)$ do.
2. Compute $p((x_i + \Delta_x, y_i + \Delta_y)|M)$, $(\Delta_x, \Delta_y) \in \{\tau, 0, \tau\} \times \{\tau, 0, \tau\}$
3. Pick up (Δ_x, Δ_y) at random by the Gibbs sampler[4].
4. repeat 2,3 until it converges.

5 Designing jump-diffusion process.

5.1 Motivations for designing the jump-diffusion process

In section (3), the mapping edges M are computed by maximizing $p(M|\Gamma)$, and in section (4) the junctions $J = \{(x_i, y_i), i = 1, 2, ..., K\}$ are computed by maximizing $p(J|M, \Gamma)$. The two separate steps work well in many nice 2D shapes, however the resulting medial axes could be suboptimal in general, because the mapping M is computed locally without considering the junctions. For example, figure 11 displays the computed M which determines three junctions of degree=3 instead of one junction of degree=6 (see figure 15), even though the latter has lower energy.

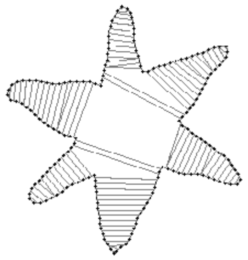


Figure 11 Short false axes appear in the mapping function.

The split and merge of junctions was considered in the FORMS system by Zhu and Yuille[27] in a top-down process which matches the medial axis graph to an object model in order to obtain the best match. In this section, we shall discuss how M and J are computed interactively by maximizing a joint probability,

$$p(J, M|\Gamma) = p(J|M, \Gamma)p(M|\Gamma) \quad \text{on a space} \quad \Omega = \Omega_M \times_{i=1}^{K(M)} D_i.$$

Equivalently we minimize the energy,

$$E(M, J|\Gamma) = \frac{1}{T_1} E_1(M|\Gamma) + \frac{1}{T_2} \sum_{i=1}^{K(M)} E_2(x_i, y_i)$$

where the number of junctions $K = K(M)$ is a function of the mapping M, and T_1, T_2 are temperatures. As it is computationally inefficient to compute J when M

is far from converging (see figure 7.a), the computation is divided into two phases. In phase I, set T_2 at a very high temperature, thus we can ignore the second term in $E(M, J|\Gamma)$ and compute M as in section (3). Then in phase II, we reduce T_2 to a low temperature and compute J together with M.

However, minimizing $E(M, J|\Gamma)$ is not a trivial problem, because the probability mass $p(J, M|\Gamma)$ is distributed over subspaces Ω_K whose dimensions changes as K varies.

$$\Omega = \cup_K \Omega_K, \quad \Omega_K = \Omega_M \times_{i=1}^{K} D_i, \qquad K = K(M).$$

This problem exists in many other computer vision problems, wherever a probability distribution involves hierarchic random variables or multiple choices of models, for example, in an image segmentation problem. In the next subsection, we briefly discuss basic issues in designing jump-diffusion processes as they have mainly been studied in statistics literatures, not in computer vision.

5.2 Basics of jump-diffusion

Stochastic jump-diffusion process was first studied by Grenander and Miller in 1994[6], and a rigorous account for the reversibility was given by Green in 1995[5].

Suppose that in an application the solution X lies in a mixture space Ω which consists of three possible subspaces $\Omega = \Omega_1(s) \cup \Omega_2(\mu, \nu) \cup \Omega_3(\xi, \rho, \eta)$ in one, two and three dimensions respectively, such as an interval, a region, and an ellipsoid in figure 12. For example, $s, (\mu, \nu), (\xi, \rho, \eta)$ could be parameters of three possible models to account for the data, such as grey level intensity, motion velocity, and color respectively. Thus the probability for X is a mixture distribution,

$$p(X) = \lambda_1 p_1(s) + \lambda_2 p_2(\mu, \nu) + \lambda_3 p_3(\xi, \rho, \eta).$$

If the objective is to compute the global optimal solution X^*, it is necessary to simulate a Markov chain Monte Carlo process for sampling $p(X)$. This stochastic

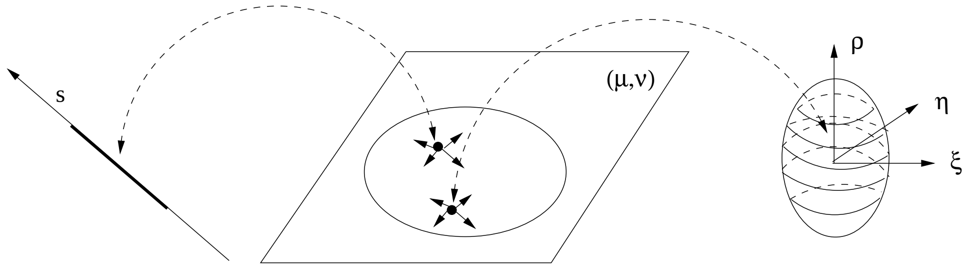


Figure 12 An illustration for probability mass defined on $\Omega = \Omega_1 \cup \Omega_2 \cup \Omega_3$, where $\Omega_1 \cup \Omega_2 \cup \Omega_3$ are subspaces that lie in 1D, 2D and 3D respectively.

process consists of two kinds of random walks, as shown in figure 12. One is diffusion, which moves X inside one subspace, e.g. in the (μ, ν)-plane. The other is jump, which switches the model from one subspace to the other. To design the jump-diffusion process, one could design many types of "proposals" for moving $X(t) \to X(t+1)$ with t being the time step. The criteria for designing the jump-diffusion process are 1). each proposal is reversible and satisfies the detailed-balance equation, and 2). the whole jump-diffusion process is ergodic and aperiodic. At each step t, one of these proposals is chosen at random, and the proposal is accepted with an acceptance probability computed from $p(X)$ as well as the proposal probabilities. The Markov chain should eventually reach an equilibrium with its status sampling the probability $p(X)$[5]. When an appropriate annealing schedule is adopted, it is guaranteed to reach the global optimal solution. It is worth noting that the stochastic process is not only important for achieving the global optimum, but is also crucial for obtaining multiple solutions and analyzing the performance of the model.

5.3 Design the jump-diffusion proposals

To minimize $E(M, J|\Gamma)$, we identified four types of proposals, which are illustrated in figure 13.

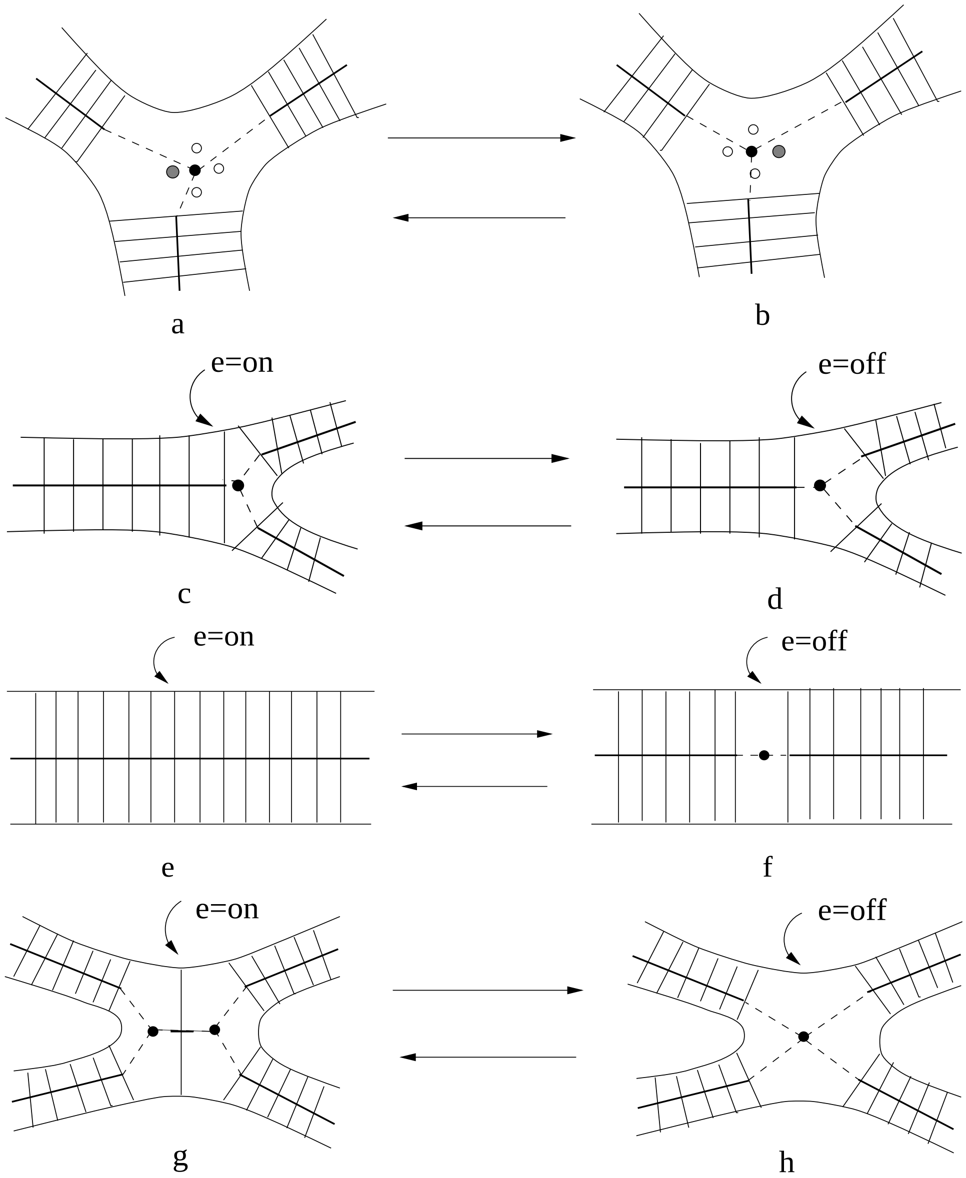


Figure 13 Four reversible proposals for designing the jumping-diffusion process.

- Type I: moving a junction (x_i, y_i) within a neighborhood D_i, see figure 13.a and 13.b. This is a diffusion process.

- Type II: flipping a mapping edge $e =$ on/off without changing the number of junctions $K(M)$, see figure 13.c and 13.d. Usually e is at the end of a pair of matched intervals. This is basically a diffusion process except that the domain D_i of the junction (x_i, y_i) associated with e changes slightly as e flips. So this step had better (doesn't have to) propose a new position for (x_i, y_i).

- Type III: flipping a mapping edge $e =$ on/off that creates or removes a junction (x, y), as is shown in figure 13.e and 13.f. This is a jump process. The proposal should include a position (x, y) for the created junction.

- Type IV: flipping a mapping edge $e =$ on/off that splits a junction (x, y) into two (x_1, y_1) and (x_2, y_2), or, reversely, merges two into one, as is shown in figure 13.g and 13.h. This is again a jump process. The proposal should also include the positions for the new junctions.

The four types of proposals are reversible and they construct an ergodic process in random combination for walking in the space Ω. Type I is implemented by the Gibbs sampler in section (4), and Types II, III, and IV flip a single mapping $e =$ on/off, but when we compute $p(e = \text{on/off}|M)$ as in Algorithm I, we should also consider the associated junction(s) simultaneously.

Take type IV as an example. The proposal switches between two configurations:

$$\begin{array}{lll} A: & \{e = \text{on}, J_1 = (x_1, y_1), J_2 = (x_2, y_2)\} \cup M^- \cup J^- & \text{(figure 13g)} \\ B: & \{e = \text{off}, J_0 = (x_0, y_0)\} \cup M^- \cup J^- & \text{(figure 13h)} \end{array}$$

where M^- and J^- denote the rest of the mapping edge set and the junction set, which are fixed during the move.

In the Metropolis-Hastings algorithm, a proposal is accepted with probability

$$\alpha(A \to dB) = \min(1, \frac{q(B \to dA)p(B)dB}{q(A \to dB)p(A)dA}) \tag{7}$$

where $q(B \to dA)$ and $q(A \to dB)$ are the proposal probabilities, which need to be decided. The proposal $A \to B$ includes flipping e and a new position (x_0, y_0) according to probability $q(x_0, y_0)$. The proposal $B \to A$ includes flipping e and two new positions (x_1, y_1) and (x_2, y_2) according to probabilities $q(x_1, y_1)$ and $q(x_2, y_2)$ respectively.

Therefore, the acceptance probability becomes,

$$\alpha(A \to dB) = \min(1, \frac{q(x_1, y_1)q(x_2, y_2)p(B)dx_1dy_1dx_2dy_2dB}{q(x_0, y_0)p(A)dx_0dy_0dA}). \tag{8}$$

It is easy to check that the dimensions of the above probabilities are well matched.

Generally speaking, in the proposal $A \to B$, the position (x_i, y_i) for the newly created junction could be randomly chosen in the domain D_i. Obviously random choices very likely lead to a high rejection rate and thus are inefficiency. We choose $q(x_i, y_i)$ to focus on the center of the domain D_i. A similar problem has been discussed in section (3) when we discuss the proposal probability from the candidate set S_i for each linelet Γ_i. Given the proposed positions, $p(e = \text{on/off}|M)$ can be computed. We shall not discuss the details of the computation but shall give the overall algorithm.

Algorithm III: A Jump-Diffusion process for Medial axis Φ

0. Partition Γ, and set T_1, T_2 to high temperatures in $E(M, J)$.
1. Anneal T_1, compute M by algorithm I.
2. Anneal T_2, compute J by algorithm II.
3. Pick up e at random.
4. If e intersects existing mappings, go to 10, otherwise decide the proposal type
5. Propose position(s) for junction(s) near e depending on the proposal type.
6. Compute $p(e = \text{on}|M)$ and $p(e = \text{off}|M)$ for e.
7. Draw a random variable $r \in [0, 1]$ with a uniform distribution
8. If $r < p(e = \text{on}|M)$, then $M \leftarrow M \cup \{e\}$, update J.

9. Otherwise $M \leftarrow M/\{e\}$, update J.
10. $count \leftarrow count + 1$.
11. If $count < threshold$ Goto 3, else stop.

In practice, M resulting from step 2 is almost always a superset of the optimal M^*, especially if we choose a large γ in $E_1(M)$, thus we only need to propose which $e \in M$ to turn off. So in step 3, it is computationally efficient if we only choose e at the end of an axis–i.e. near the junctions. If our goal is to compute the minimum of $E(M, J)$, not to sample the positions of the junction at each step, an efficient proposal is to turn off an axis that consists of only a small number of mapping edges. Figure 11 shows two of such short axes each of which has only two mapping edges.

6 Experiments

We demonstrate some typical medial axes in figure 14 and figure 15. In figure 14.a, a perfect "T"-junction is recovered. In figure 14.b there are five long axes for the rectangular parts, and they are virtually connected by a junction. The computed axes in figure 14 are the most favorable among many other possible axes. Intuitively, the perception of axes largely depends on the constructing parts of object shape and the junctions joining the parts. The axes in figure 14 imply butt joints between rectangular parts, while the axes in figure 1 assume hinge joints. Some axes are also obtained for the real shapes with nice junctions. For example, the six axes in the flower shape joined at a single point and they are aligned in three groups; in the hand shape, the fingers and the palm axes joined at one point, and a nice "T"-junction forms for the thumb-palm junction.

7 Discussion

In this paper, the computation of medial axes is posed as a statistical inference problem, not a deterministic transform by maximum disks. Medial axes are defined as modes of a probability $p(M, J|\Gamma)$ in a space Ω. A stochastic algorithm is proposed for sampling $p(M, J|\Gamma)$ and for minimizing the energy functional through a rigorous account of a jump-diffusion process. This jump-diffusion process realizes

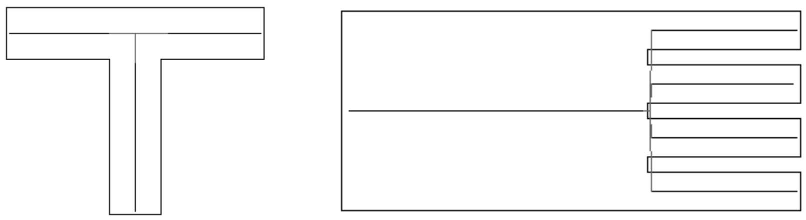


Figure 14 The stochastic medial axes for two synthetic shapes

the interactions between M and J, which is often known as the *bottom-up/top-down* computation in vision.

The algorithm is limited to computing M, J from Γ, i.e.

$$\Gamma(s) \rightarrow M(\mathrm{or}\, \phi(s)) \Longleftrightarrow J.$$

Ideally we shall complete the loop by computing medial axes as well as Γ from real textured images $\mathbf{I}^{\mathrm{obs}}$, i.e.

$$\mathbf{I}^{\mathrm{obs}} \rightarrow \Gamma(s) \Longleftrightarrow M(\mathrm{or}\, \phi(s)) \Longleftrightarrow J.$$

This will involve more complex computation in a bottom-up/top-down manner. The robustness problem of the medial axis shall be solved if a smoothness prior model is used on the boundary Γ. We expect that this stochastic algorithm should also be useful in perceptual grouping at the middle-level vision and in detecting symmetric parts of open curves. The Markov mapping graphs and the stochastic algorithms in this paper have been utilized in learning and sampling probability models of shape $p(\Gamma)$ in a companion paper[30].

Furthermore, the Markov property of our medial axis definition and the stochastic nature of the algorithm make it plausible for massive parallel computation and simultaneous extraction of medial axes during the segmentation of images – as was suggested by experiments in psychology and neurophysiology[10, 11]. However, many issues have to be resolved before applying the algorithm to fragmented curves.

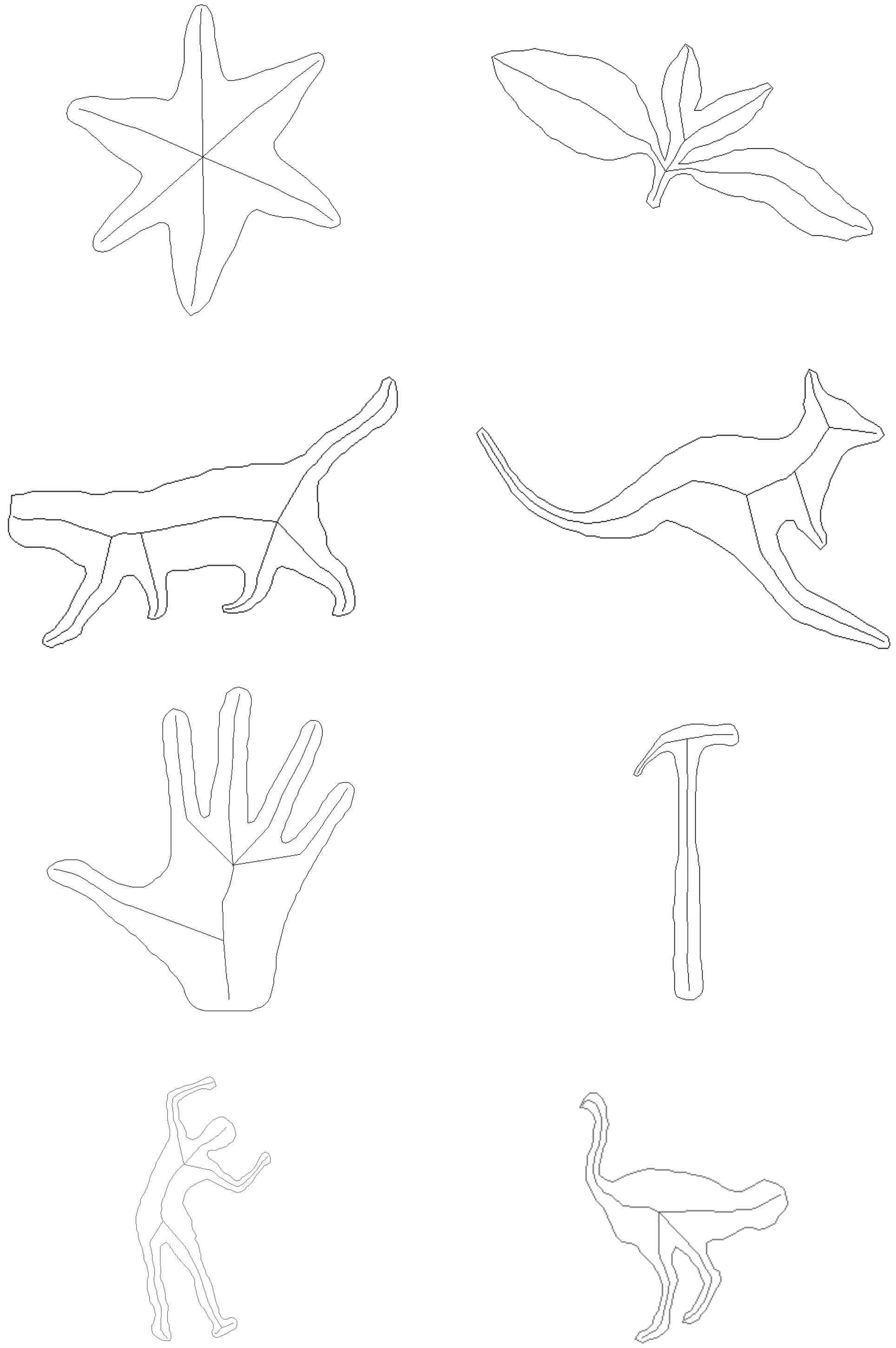


Figure 15 The stochastic medial axes for some real shapes.

Acknowledgment: The author is supported by a grant from the ARO Center for Image Sciences DAAH-04-95-1-0494, and a Microsoft gift. The author would like to thank David Mumford, Y. N. Wu and Z. Y. Chi for stimulating discussions.

References

[1] H. Blum, “Biological shape and visual science”. *J. of Theoretical Biology*, 33, pp205-287, 1973.

[2] J. Brady and H. Asada, “Smooth local symmetries and their implementations”. *Int. J. of Robotics Reg.* 3(3), 1984.

[3] J. L. Crowley, “A representation for shape based on peaks and ridges in the difference of low-pass transform”. *IEEE Trans. Patt. Anal. Mach. Intell.*. PAMI-6(2). March 1984.

[4] S. Geman and D. Geman. “Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images”. *IEE Trans. PAMI* **6**. pp721-741, 1984.

[5] P. J. Green, “Reversible jump Markov chain Monte Carlo computation and Bayesian model determination”, *Biometrika*, 82, 4, pp. 711-32, 1995.

[6] U. Grenander and M. Miller, “Representation of knowledge in complex systems”, *Jorunal of Royal Statistics Society*, B 56, pp 97-109, 1994.

[7] M. Kass, A. Witkin and D. Terzopoulos. “Snakes: Active Contour Models”, *Proc. Int. Conf.on Computer Vision ICCV87.* London, England. 1987.

[8] M. F. Kelley and M. D. Levine, “Symmetry to representation: the abstraction of object information from images”, *TR-CIM-94-12*, Centre for Intelligent Machines, McGill Univ. 1995.

[9] B. B. Kimia, A. R. Tannenbaum, and S. W.Zucker, “Shape, shocks and deformations I: the components of 2D shape and the reaction-diffusion space”, *Int'l J. of Computer Vision*, 15, pp189-224, 1995.

[10] I. Kovacs and B. Julesz. "Perceptual sensitivity maps within globally defined visual shapes", *Nature*, 1994.

[11] T. S. Lee, D. B. Mumford and S.C. Zhu. "Sensitivity of V1 neurons to globally defined shapes", *Proc. 5th Int'l Conf. on Computational Neurosciences*, 1996.

[12] F. Leymaric and M. D. Levine, "Simulating the grass-fire transform using an active contour model", *IEEE Trans. PAMI* vol.14, no1, January,1992.

[13] M. Leyton, *Symmetry, Causality, Mind.* MIT Press. Cambridge, Mass. 1992.

[14] S. Li, "Object recognition from range data", IVC, October, 1992.

[15] S. Li, "Markov random field modeling in computer vision", Springer-Verlag, 1995.

[16] T. Liu, D. Geiger, and R. Kohn, "Representation and self-similarity of shapes", *Proc. Int'l Conf. on Computer Vision*, Bambay, India, 1998.

[17] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller,and E. Teller, "Equations of state calculations by fast computing machines." *J. Chem. Phys.* 21, 1087-92. 1953.

[18] D. Mumford and J. Shah. "Optimal approximations by piecewise smooth functions and associated variational problems." *Comm. Pure Appl. Math.*, 42, pp 577-684. 1989.

[19] R. Navatia & T.O. Binford, "Description and recognition of curved objects". *Artificial Intelligence*, 8, pp77-98, 1977.

[20] R. L. Ogniewicz, *Discrete Voronoi skeleton.* Hartung-Gorre,1993.

[21] G. Sapiro, "Affine invariant medial axis and skew symmetry", Proc. Int'l Conf. Computer Vision, Bambay, India, 1998.

[22] S. Tari, and J. Shah, "Local symmetries of shapes in arbitrary dimension", *Proc. Int'l Conf. on Computer Vision*, Bambay, India, 1998.

[23] K. Siddiqi, A. Tannenbaum, and S. Zucker, "Hyperbolic smoothing of shapes", *Proc. Int'l Conf. on Computer Vision*, Bambay, India, 1998.

[24] Y. F. Tsao and K. S. Fu, "Stochastic skeleton modeling of objects", *CVGIP*, vol. 25, pp.384-370, 1984.

[25] M. Wright, R. Cipolla, and P. Giblin, "Skeletonisation using an extended Euclidean distance transform", *Image and Vision Computing*, 1995.

[26] S. C. Zhu and A. L. Yuille, "Region Competition: unifying snakes, region growing, and Bayes/MDL for multi-band image segmentation", *IEEE Trans. PAMI*, Vol.18, No.9, Sept., 1996.

[27] S. C. Zhu and A. L. Yuille, "FORMS: a Flexible Object Recognition and Modeling System", *Int'l Journal of Computer Vision*, Vol.20, No.3, Dec, 1996.

[28] S. C. Zhu and D. B. Mumford, "Prior learning and Gibbs reaction-diffusion", *IEEE Trans. on PAMI*, vol.19, no.11, Nov. 1997.

[29] S. C. Zhu, "Stochastic computation of medial axis in the Markov random fields", *Proc. of Int'l Conf. on Computer Vision and Pattern Recognition*, Santa Barbara, CA, 1998.

[30] S. C. Zhu , "Embedding Gestalt laws in the Markov random field", *IEEE Trans. on PAMI*, Vol. 21, No. 11, November, 1999.

[31] S. W. Zucker, "Region growing: childhood and adolescence", *Computer Vision, Graphics and Image Processing*, Vol. 5, 1976.