A Robust Divide and Conquer Algorithm for Progressive Medial Axes of Planar Shapes

Yong-Jin Liu, *Member, IEEE*, Cheng-Chi Yu, Min-Jing Yu, Kai Tang, and Deok-Soo Kim

Abstract—The medial axis is an important shape representation that finds a wide range of applications in shape analysis. For large-scale shapes of high resolution, a progressive medial axis representation that starts with the lowest resolution and gradually adds more details is desired. In this paper, we propose a fast and robust geometric algorithm that computes progressive medial axes of a large-scale planar shape. The key ingredient of our method is a novel structural analysis of merging medial axes of two planar shapes along a shared boundary. Our method is robust by separating the analysis of topological structure from numerical computation. Our method is also fast and we show that the time complexity of merging two medial axes is $O(n \log n_v)$, where n is the number of total boundary generators, n_v is strictly smaller than n and behaves as a small constant in all our experiments. Experiments on large-scale polygonal data and comparison with state-of-the-art methods show the efficiency and effectiveness of the proposed method.

Index Terms—Progressive medial axes, shape hierarchy and evolution, topology-oriented algorithm, divide and conquer algorithm

1 Introduction

The medial axis is the locus of all points that have at least two closest points on the shape boundary. After it was first introduced in 1960s, the medial axis has found rich applications in a great diversity of disciplines [1], such as free-form shape design and animation in computer graphics, motion planning in robotics, shape analysis and recognition in computer vision, data compression in image processing, tool path generation and finite element mesh generation in CAD/CAM, etc.

Nowadays, large-scale shape data of high resolution is emerging. For example, in the topographical data of U.S. geological survey, the boundary of each polygonal shape of six continents has more than 100 K vertices (Fig. 1 and Table 2). Although many classic and state-of-the-art medial axis computation methods had been proposed (e.g., [2], [3], [4], [5], [6], [7]), two challenges still exist. First, serious degenerate cases frequently appear in large-scale data. Therefore a robust yet fast geometric algorithm is needed. Secondly, it often takes a long time to compute the medial axis of a large-scale planar shape. Users usually cannot wait for a long time until the result suddenly appears. Therefore, a progressive medial axis representation is much desired.

- *Y.J. Liu, C.C. Yu and M.J. Yu are with the TNList, the Department of Computer Science and Technology, Tsinghua University, Beijing, China. E-mail: liuyongjin@tsinghua.edu.cn, yccbupt@gmail.com, yumj14@mails.tsinghua.edu.cn.*
- *K. Tang is with the Department of Mechanical and Aerospace Engineering, The Hong Kong University of Science and Technology, Hong Kong, China. E-mail: mektang@ust.hk.*
- *D.S. Kim is with the Voronoi Diagram Research Center and School of Mechanical Engineering, Hanyang University, Seoul, Korea. E-mail: dskim@hanyang.ac.kr.*

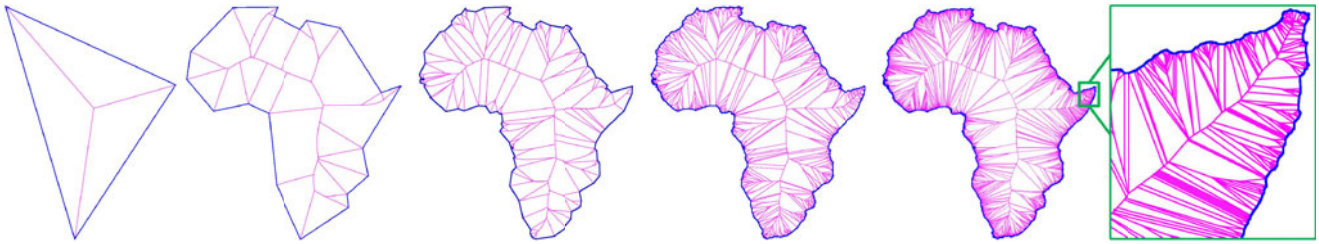
Manuscript received 21 Apr. 2015; revised 12 Oct. 2015; accepted 13 Dec. 2015. Date of publication 23 Dec. 2015; date of current version 17 Oct. 2016. Recommended for acceptance by M. Botsch.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2015.2511739

In this paper, we propose a fast and robust geometric algorithm that computes progressive medial axes of a large-scale planar shape. The key ingredient of our method is a topology-oriented algorithm that merges medial axes of two planar shapes along a shared boundary. Our method can work with any third-party shape decomposition methods as long as the resulting sub-shapes share some boundaries and their interiors do not intersect. In particular, we offer the following two contributions in this paper.

The first contribution is that our method can quickly and robustly compute the medial axis of a large-scale shape. Notably there are three classes of robust geometric algorithms in literature. The first class is the exact computation that uses either modular or multi-precision integer to obtain correct geometric predicates [8]. However, the implementation of exact computation is usually sophisticated and its performance is very time consuming. For example, for the large-scale polygonal shape with 204,545 vertices (Fig. 1), a state-of-the-art implementation of exact computation in Computational Geometry Algorithm Library[1] (CGAL) using `CORE::Expr` as the predicate kernel [9] runs for more than one week and still cannot output its medial axis. The second class uses floating-point filters (FPF) based on an analysis of numerical error bounds (e.g., [10], [11]). This class of methods compute the predicates first in floating arithmetic. Only when the result is unreliable, exact arithmetic is used secondly. For the same large-scale shape in Fig. 1, an implementation of a floating-point filter method in CGAL [7] runs 5.4 hours to obtain its medial axis. The third class is the topology-oriented method [12], [13], [14]. In this class of methods, the basic part of a geometric algorithm is described in terms of combinatorial and topological computation primarily, and numerical computation that is compatible with the topological structure is used as second information. To our best knowledge, our proposed method is the first topology-oriented method for

1. https://www.cgal.org/

1077-2626 © 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.



(a) 3 vertices (b) 20 vertices (c) 100 vertices (d) 500 vertices (e) 4,076 vertices

Fig. 1. Progressive medial axes of a large-scale polygonal shape for the Africa model. The original polygon has 204,545 vertices. To compute its medial axis, the exact computation in CGAL runs more than one week and still cannot output the result. The floating-point filter implementation in CGAL takes 5.4 hours. As a comparison, our topology-oriented method only takes 14.8 minutes. All the results are obtained on a PC (Intel(R) Core (TM) I7920 CPU 2.67GHz). Due to limited resolution, Only polygons with 3, 20, 100, 500, and 4,076 vertices are illustrated. At any time during the process, a user can view a progressive medial axis on-the-fly and has a better user experience.

computing the medial axis. For the same shape in Fig. 1, our method takes only 14.8 minutes[2] to obtain the result. More experiments are presented in Section 6 and results show the efficiency of our method.

The second contribution is that we propose a progressive medial axis representation of large-scale shapes. For a large-scale shape with hundreds of thousands of vertices, state-of-the-art algorithms may take a long time to compute its medial axis (e.g., at least 14.8 minutes to compute the medial axis of the shape in Fig. 1). That means, a user has to wait for a long time until the result suddenly appears. To improve users' experience, we first decompose the original large-scale shape using any third-party decomposition methods (e.g., contour evolution [15] or segmentation-based decomposition [16]). Then starting from the most simplified shape such as a triangle, we compute its medial axis and progressively update the medial axis in a refinement process (Fig. 1). During the refinement process, a user can at any time view a progressive refinement of both the shape and its medial axis on-the-fly. Two progressive methods, 2D region growing and 1D boundary evolution, are proposed in this paper.

2 Preliminaries

For a set X, denote by $|X|$, $\overline{X}$, $\mathring{X}$ and ∂X, respectively, the cardinality, closure, interior and boundary of X. In this paper, a shape Ω is a bounded, connected open set in $\mathbb{R}^2$ and its boundary $\partial\Omega = \overline{\Omega} \setminus \Omega$ consists of a finite number of mutually disjoint simple closed curves. Each closed curve consists of a finite number of real analytic curve segments.

In $\partial\Omega$, the simple closed curve that bounds the unbounded, connected component in $\mathbb{R}^2 \setminus \Omega$ is called the outer loop of Ω, and the remaining curves in $\partial\Omega$ are called the inner loops. The number h of inner loops is called the genus of Ω (we also say Ω has h holes). A shape Ω is *simple* if it has no holes; otherwise it is multiply connected.

The segments in each loop of $\partial\Omega$ are oriented if when walking from their starting points to end points, the interior of Ω always lies to the left side. For each loop, its constituting segments are separated by vertices where the unit tangent vector field of the loop can be discontinuous. A vertex is called *reflex* if its interior angle is larger than π. Otherwise, the vertex is called convex.

2. There are totally 2,034,175 critical points appeared in separators and 6,102,525 updates in mating generator lists in our algorithm; see Sections 3-6 for details.

Definition 1. *A shape boundary $\partial\Omega$ consists of edges and reflex vertices. An edge can be either linear or non-linear. A non-linear edge is called general if it is not a circular arc. The point of local positive maximal curvature in a general non-linear edge can only occur at the endpoint of that edge. Each of edges and reflex vertices is parameterized in the domain $(0, 1)$, where edges are based on a usual arc-length parameterization while reflex vertices are based on the angle range spanned by two inward unit normal vectors of two incident edges at that vertex (Fig. 2).*

When a polygon is represented by one or more B-spline curves, by Definition 1, the B-spline curves are separated at points of local positive maximal curvature and decomposed into a set of non-linear edges.

For any point $x \in \Omega$, denote by $\Lambda(x)$ the set of closest boundary points of x, i.e., $\Lambda(x) = \{p \in \partial\Omega : dist(x, p) = dist(x, \partial\Omega)\}$, where $dist(x, \partial\Omega)$ is the minimal Euclidean distance between x and $\partial\Omega$.

Definition 2. *The medial axis $M(\Omega)$ of Ω is the set of points in Ω which has at least two closest boundary points:*

$$M(\Omega) = \{x \in \Omega : \Lambda(x) \text{ contains at least two distinct points}\}$$

Elements in $M(\Omega)$ are called medial points. If $\Lambda(x)$ is countable and $|\Lambda(x)| \geq 3$, x is called a branch point of degree $|\Lambda(x)|$. A branch point is regular if its degree is three, otherwise it is irregular.

An open disk $D \subseteq \Omega$ is said to be *maximal* if every disk in Ω that contains D equals D.

Definition 3. *The skeleton $Sk(\Omega)$ of Ω is the set of centers of maximal disks in Ω.*

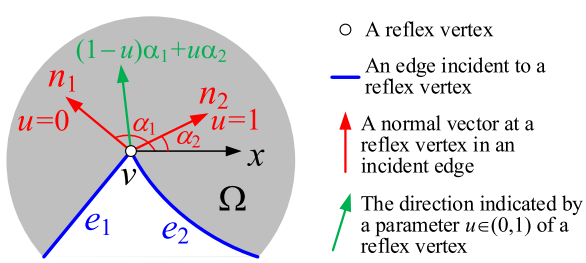
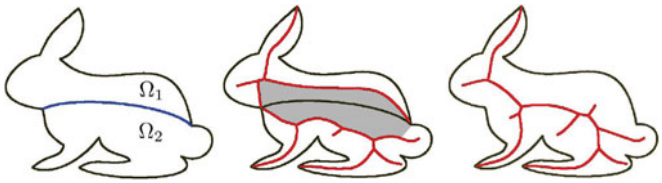


Fig. 2. The parameterization of a reflex vertex v. e_1 and e_2 are two incident edges of v. The shaded area is inside the shape Ω. n_1 and n_2 are normal vertices of two incident edges at v respectively. In the predefined orientation of $\partial\Omega$, e_1 is before v and e_2 is after v. The angle between n_1 (or n_2) and x axis is α_1 (or α_2). The parameter domain $(0, 1)$ of v is mapped to the unit vectors, where $u = 0$ corresponds to n_1, $u = 1$ to n_2, and $u \in (0, 1)$ to the vector indicated by $(1 - u)\alpha_1 + u\alpha_2$.



(a) Two sub-shapes (b) Two medial axes (c) Merged medial axis

Fig. 3. When two sub-shapes (a) merged into one ($\Omega_1 \cup \Omega_2$) by removing the shared boundary, compared to the union of two medial axes of sub-shapes (b), the medial axis of the merged shape (c) is changed locally inside the shaded area in (b).

$Sk(\Omega)$ is different from $M(\Omega)$ only at some skeleton points whose closest boundary points are isolated points of positive maximal curvature or convex vertices. It was shown that $M(\Omega) \subseteq Sk(\Omega)$ and $\overline{M}(\Omega) = \overline{Sk}(\Omega)$ (page 383 in [17]), and two examples are illustrated in the supplemental material.

$\overline{M}(\Omega)$ is a connected planar graph with finitely many vertices and edges, as long as there are finitely many edges and reflex vertices in $\partial\Omega$ [18]. We call an edge in $\overline{M}(\Omega)$ a *medial curve*, which is a single trimmed bisector of two generators:

- For any two generators from points, line segments and circular arcs, their bisector is just a line or a conic arc [19].
- The bisector of a point and a polynomial (or rational) curve segment can be parameterized exactly in a rational Bézier form [20].
- The bisector of two polynomial (or rational) curve segments other than conics does not in general admit exact parametrization. Procedures for computing ordered sequences of discrete data to approximate the true bisector with any prescribed tolerance were proposed in [2], [21], [22].

In Sections 3 to 5, a shape Ω is assumed to be simple. If Ω is not simple, an augmented domain technique [3] is introduced in Section 5.3 to convert Ω into a simple shape.

3 Overview of the Proposed Method

Given a large-scale shape Ω, let $\Omega = \{\Omega_1, \Omega_2, \ldots, \Omega_m\}$ be a shape decomposition by any third-party methods, such that $\Omega = \cup_{j=1}^{m}\Omega_j$ and $\mathring{\Omega}_j \cap \mathring{\Omega}_k = \varnothing$, $j \neq k$. Then the intermediate shapes $\cup_{j=1}^{i}\Omega_j$, $i = 1, 2, \ldots, m$, provide a progressive shape representation of Ω. Our method is designed to quickly and robustly compute $M(\cup_{j=1}^{i}\Omega_j)$ in a local updating fashion.

A key observation is that if two subshapes $\cup_{j=1}^{i}\Omega_j$ and Ω_{i+1} always share some boundaries and their interiors do not intersect, then the change of medial axes by merging two subshapes is local around the shared boundary (Figs. 3b and 3c). We extend our preliminary work on dynamic medial axes [23] and propose a symbolic representation called *mating generator list* to characterize the topological structure of medial axis (Section 4). We show that for merging medial axes, this topological representation can separate combinatorial structure updating from numerical computation, and thus a topology-oriented robust implementation is feasible (Section 5). Finally two progressive methods based on third-party shape decompositions are presented (Section 6).

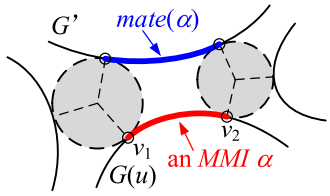


Fig. 4. A maximum mating interval (shown in red) α on G and its $mate(\alpha)$ (shown in blue) on G'.

4 Mating Relation in Medial Axis

Our topological structure analysis for merging medial axes along shared boundary relies on a mating relation as defined below. To ease reading, the proofs of main results are presented in the appendix.

Definition 4. *Two points p and q in $\partial\Omega$ are said to mate to each other if there exists an $x \in M(\Omega)$ such that $\{p, q\} \subseteq \Lambda(x)$. x is called the medial point of the mating pair $\{p, q\}$ and p is called a mating point[3] of q. Denote the set of mating points of q as $mate(q)$. If there is only one point p in $mate(q)$, we also write $p = mate(q)$. If $mate(q)$ contains more than one point in a circular arc C_{arc}, all these mating points in C_{arc} are treated as a single point C_{arc}.*

Definition 5. *Let E be an edge or a reflex vertex in $\partial\Omega$. E is called a generator if the mating points of E contains at least one point not in E.*

If the mating points of a circular arc C_{arc} in $\partial\Omega$ are C_{arc} itself only, by Definition 5, this circular arc C_{arc} is no longer a generator. By Definition 1, the point of local positive maximal curvature cannot occur inside a generator. Then the mating points corresponding to a medial point in $M(\Omega)$ always belong to different generators in $\partial\Omega$ [1].

Definition 6. *Let G be a generator in $\partial\Omega$. A continuous portion $\alpha = [v_1, v_2]$ defined in the parameter domain $(0, 1)$ of G is called a maximum mating interval (MMI) on G, if*

1) *the mating points $mate(G(u))$, $u \in [v_1, v_2]$, belong to a same generator G', $G' \neq G$, and*
2) *for any sufficiently small value $\varepsilon > 0$, the mating points $mate(G(u))$, $u \in [v_1 - \varepsilon, v_2]$ or $u \in [v_1, v_2 + \varepsilon]$, lie in at least two different generators G' and G''.*

Denote by $mate(\alpha)$ the set of mating points on G' of an MMI α on G.

Fig. 4 illustrates Definition 6. A simple argument can show that $mate(\alpha)$ of an MMI α must also be an MMI on another generator G'.

Definition 7. *For an MMI α on G, let $\alpha' = mate(\alpha)$ be an MMI on G'. α' is called a mating MMI of α and G' is called a mating generator of G. Furthermore, $\{\alpha, \alpha'\}$ and $\{G, G'\}$ are called a mating pair of MMIs and generators, respectively.*

Lemma 1. *Suppose that Ω is simple. On a mating pair of generators $\{G, G'\}$, there exists exactly one mating pair of MMIs.*

3. The mating point relation needs not to be a one-to-one mapping. E.g., corresponding to a branch point, a boundary point has two or more mating points. In [1], the mating point is called the medial involute. In this paper, we use consistent notations of mating points, mating intervals and mating generators.

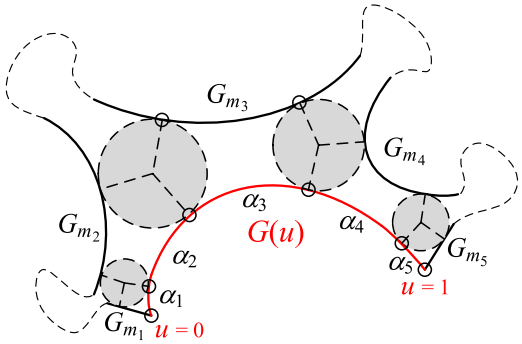


Fig. 5. The mating generator list $\Phi(G)$ of the generator $G : \Phi(G) = \{G_{m_1}, G_{m_2}, G_{m_3}, G_{m_4}, G_{m_5}\}$, $G_{m_5} \prec_G G_{m_4} \prec_G G_{m_3} \prec_G G_{m_2} \prec_G G_{m_1}$.

By Lemma 1, a mating pair of generators $\{G_i, G_j\}$ can correspond to only one medial curve and we denote it by $m_{ij} = m(G_i, G_j)$.

A consequence of Lemma 1 is the preservation of inverse ordering of mating generators with respect to their ordering on $\partial\Omega$. We first define an *in-front* relation that induces an ordering on the generators in $\partial\Omega$. Let a, b and c be three distinct points in $\partial\Omega$ with a consistent orientation as specified in Section 2. Point b is said to be *in front* of c with respect to a, denoted as $b \prec_a c$, if when one starting at a walks along the loop with a predefined orientation, point b is encountered before c. Further, if points a, b and c lie on three distinct generators G_a, G_b and G_c, respectively, then G_b is said to be *in front* of G_c with respect to G_a, denoted as $G_b \prec_{G_a} G_c$.

Definition 8. *Let $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ be the set of ascending ordered MMIs with respect to the parameter u on a generator $G(u)$ and $\{G_{m_1}, G_{m_2}, \ldots, G_{m_r}\}$ be the mating generators that correspond to $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$. Induced from the order in $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$, the list $\Phi(G) = \{G_{m_1}, G_{m_2}, \ldots, G_{m_r}\}$ is also ordered (reversely in the second subscript). We call $\Phi(G)$ the mating generator list of G.*

Fig. 5 illustrates a mating generator list $\Phi(G)$.

Lemma 2. *$G_i \in \Phi(G_j)$ if and only if $G_j \in \Phi(G_i)$.*

Lemma 3. *Let G be a generator in a simple shape Ω. All the generators in $\Phi(G)$ are distinct.*

Lemma 4 (Inverse order preservation lemma). *Let the generators of a simple shape Ω be ordered using the in-front relation with respect to a generator G, i.e., $G_1 \prec_G G_2 \prec_G \cdots \prec_G G_n$. In the mating generator list $\Phi(G) = \{G_{m_1}, G_{m_2}, \ldots, G_{m_r}\}$, $G_{m_j} \prec_G G_{m_i}$, $i, j \le r \le n$, if and only if $j > i$.*

For a simple shape Ω with ordered generators $G_1 \prec_G G_2 \prec_G \cdots \prec_G G_n$, we symbolically represent $M(\Omega)$ by $\Phi(\Omega) = \{\Phi(G_1), \Phi(G_2), \ldots, \Phi(G_n)\}$. $\Phi(\Omega)$ is a topological description of $M(\Omega)$ and give a solution to a topology-oriented implementation in Section 5 for robustly computing the medial axis. A simple example is presented below.

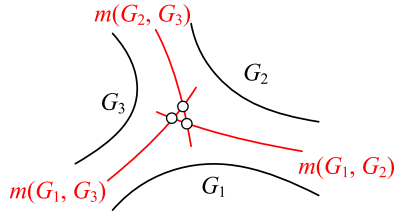


Fig. 6. Branch point identification with topological consistency checking.

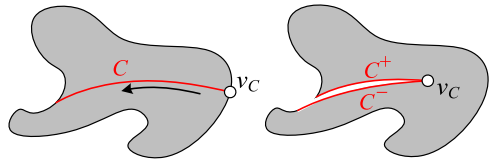


Fig. 7. The shared boundary C as a special (combinatory) generator $C' = \{C^+, C^-, v_C\}$.

A simple topology-oriented branch point identification is illustrated in Fig. 6: if G_2 follows G_3 in $\Phi(G_1)$ (i.e., $\{G_3, G_2\} \subseteq \Phi(G_1)$), then there must be a branch point as the intersection of two medial curves $m(G_1, G_3)$ and $m(G_1, G_2)$. The degree of this branch point can be quickly determined by investigating $\Phi(G_2)$ or $\Phi(G_3)$. If $G_3 \in \Phi(G_2)$ (or $G_2 \in \Phi(G_3)$), then the degree must be three. Note that the three intersection points $m(G_1, G_2) \bigcap m(G_1, G_3)$, $m(G_1, G_2) \bigcap m(G_2, G_3)$ and $m(G_1, G_3) \bigcap m(G_2, G_3)$, are very likely to be different due to numerical imprecision. For a topology-consistent answer, these three different intersection points all pertain to the same regular branch point that can be represented by their arithmetic average.

Given a triangle $\triangle$, it is easy to specify $\Phi(\triangle)$. Let $G_1 \prec_{G_1} G_2 \prec_{G_1} G_3$ be three ordered boundary generators of a triangle. Then $\Phi(G_1) = \{G_3, G_2\}$, $\Phi(G_2) = \{G_1, G_3\}$ and $\Phi(G_3) = \{G_2, G_1\}$. Starting from triangles, we apply the topology-oriented merging method presented in the next section to compute $M(\Omega)$.

5 Topology-Oriented Merging of Medial Axes

Let Ω_1 and Ω_2 be two sub-shapes that touch each other at a shared boundary $C = \partial\Omega_1 \bigcap \partial\Omega_2 \neq \emptyset$, $\mathring{\Omega}_1 \bigcap \mathring{\Omega}_2 = \emptyset$. In this paper, C is called a *separator* and we consider the case that C is a simple open curve, that is, C is not self-intersecting and can be parameterized over the interval $u = (0, 1)$ by a one-to-one mapping function $C(u)$ (Fig. 7 left).

Denote one endpoint of C by v_C. Without loss of generality, assume v_C initially having parameter $u = 1$. To obtain $M(\Omega_1 \bigcup \Omega_2)$, we shrink C by moving v_C from $C(1)$ to $C(0)$. Let $\Omega(u)$ symbolize the shape in which the position of moving v_C is specified by the parameter u, $0 < u < 1$. To ensure that $\Omega(u)$ is a correct shape, the separator $C(u)$ is treated as a special generator due to its orientation: it is a double edge with both sides facing the interior of $\Omega(u)$. We interpret $C(u)$ as a wedge in $\Omega(u)$ with zero width, as illustrated in Fig. 7 right. We denote the two sides of C by C^+ and C^-, and $C = \{C^+, C^-, v_C\}$. Let $\Phi(C) = \{\Phi(C^+), \Phi(C^-), \Phi(v_C)\}$.

5.1 Topological Structure of $M(\Omega(u))$

Note that the difference between $M(\Omega_1 \bigcup \Omega_2)$ and $M(\Omega_1) \bigcup M(\Omega_2)$ is strictly localized within those medial curves formed by the composite generator $C = \{C^+, C^-, v_C\}$ with its mating generator list $\Phi(C)$.

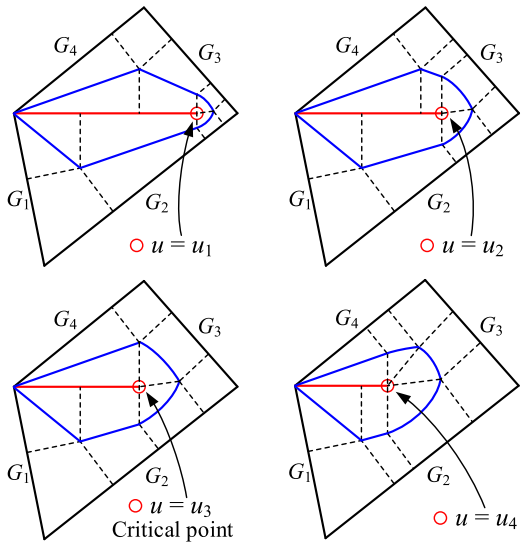


Fig. 8. The equivalence relation $\sim$ in the parameter domain $(0,1)$ of the separator C. The separator C is shown in red and the medial curves formed by C with $\Phi(C)$ are shown in blue. See text for detailed description of topological changes in the mating generator list $\Phi(C)$.

Definition 9. *Two medial axes $M(\Omega(u_1))$ and $M(\Omega(u_2))$ are said to be topologically equivalent to each other if the two separators $C(u_1)$ and $C(u_2)$ have the identical mating generator list $\Phi(C)$. An equivalence relation on the parameter domain $(0,1)$ is defined as $u_i \sim u_j$, $u_i, u_j \in (0,1)$, if and only if $M(\Omega(u_i))$ and $M(\Omega(u_j))$ are topologically equivalent.*

An example is illustrated in Fig. 8, where $u_1 \sim u_2$ and $u_1 \nsim u_4$; u_3 corresponds to a critical point (to be defined below):

- When $u = u_1$, $\Phi(C^+(u_1)) = \{G_4, G_3\}$, $\Phi(v_C(u_1)) = \{G_3, G_2\}$ and $\Phi(C^-(u_1)) = \{G_2, G_1\}$.
- When $u = u_2$, $\Phi(C^+(u_2)) = \{G_4, G_3\}$, $\Phi(v_C(u_2)) = \{G_3, G_2\}$ and $\Phi(C^-(u_2)) = \{G_2, G_1\}$.
- When $u = u_4$, $\Phi(C^+(u_4)) = \{G_4\}$, $\Phi(v_C(u_4)) = \{G_4, G_3, G_2\}$ and $\Phi(C^-(u_4)) = \{G_2, G_1\}$.

The equivalence relation $\sim$ in Definition 9 induces a quotient space in the parametric domain $(0,1)$; i.e., $(0,1)$ is partitioned into a set of equivalent classes $[u] \in (0,1)/\sim$. We characterize the transition between equivalent classes by critical points (to be defined below) and accordingly, the structure merging of $M(\Omega_1) \bigcup M(\Omega_2)$ into $M(\Omega_1 \bigcup \Omega_2)$ can be determined by identifying all the critical points in the parametric domain $(0,1)$ for C. One real-world example is shown in Fig. S2 in the supplemental material.

Due to the continuous shrinking of C, between two topologically inequivalent parameters, there must exist at least a parameter as the transition point, e.g., u_3 in Fig. 8.

Definition 10. *Let X be one of C^+, C^- and the endpoint v_C. The parameter $u = u_c \in (0,1)$ is a critical point of C if and only if there is a generator G and a sufficiently small value $\varepsilon > 0$, such that*

- *G is in $\Phi(X(u))$ for $u \in (u_c, u_c + \varepsilon)$ but not for $u \in (u_c - \varepsilon, u_c)$, or*
- *G is in $\Phi(X(u))$ for $u \in (u_c - \varepsilon, u_c)$, but not for $u \in (u_c, u_c + \varepsilon)$.*

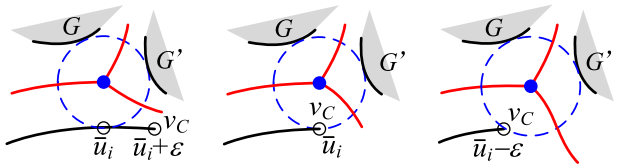


Fig. 9. Topological structure updating when traversing a switch critical point $\overline{u}_i$ in $\Phi(C^+)$, in which ε is a sufficiently small value.

Note that for a critical point u_c, there exists a corresponding irregular branch point[4] in $M(\Omega(u_c))$.

Lemma 5. *Let $\Pi(u)$ represent the union of all mating relations in the shape $\Omega(u)$. Suppose $0 < u_2 < u_1 < 1$. A mating pair of points $\{p, q\} \in \Pi(u_2)$ does not belong to $\Pi(u_1)$, if and only if there exist two distinct parameters $u_p, u_q \in (u_2, u_1]$ such that $\{p, C(u_p)\}$ and $\{q, C(u_q)\}$ are two mating pairs of points in $\Pi(u_1)$.*

There are two important implications to Lemma 5. First, when the separator C is shrunk by moving $v_c(u)$ from $u = 1$ to $u = 0$, no new generators enter the mating generator lists $\Phi(C^+)$ and $\Phi(C^-)$. Secondly, we have the following corollary.

Corollary 1. *If a critical point is traversed during the shrinking process, one of the two following cases occurs:*

Case 1. *A generator leaves $\Phi(C^+)$ or $\Phi(C^-)$, and a generator enters $\Phi(v_C)$,*

Case 2. *A generator leaves $\Phi(v_C)$.*

According to Corollary 1, there are two types of critical points (switch and vanishing critical points), and they are analyzed in Sections 5.2.1 and 5.2.2, respectively.

5.2 Updating Topological Structure in Merging Process

The critical points on the separator $C(u) = \{C^+, C^-, v_C\}$ are classified into two types, one is related to $\{C^+, C^-\}$ and the other is related to v_C.

5.2.1 Topological Structure Updating in $\Phi(C^+) \bigcup \Phi(C^-)$

We analyze the case with C^+ below and the case with C^- can be analyzed in the same way. Let $S_{C^+} = \{\overline{u}_1, \overline{u}_2, \ldots, \overline{u}_m\}$ be the set of delimiting points of MMIs on C^+, sorted in the descending order of u values. By Definitions 6 and 10, each delimiting point $\overline{u}_i \in S_{C^+}$ is a critical point and we refer to its type as *switch critical point*.

Refer to Fig. 9. When v_C moves across a switch critical point $\overline{u}_i$, the boundary generator G' that offers mating points to $(\overline{u}_i, \overline{u}_{i-1})$ leaves $\Phi(C^+)$ and the boundary generator G that offers mating points to $(\overline{u}_{i+1}, \overline{u}_i)$ enters $\Phi(v_C)$. Note that $\overline{u}_{i+1} < \overline{u}_i < \overline{u}_{i-1}$. When $\overline{u}_i$ is being traversed, the locally topological change in $\Phi(\Omega)$ is summarized in Algorithm 1: Update_Switch($\overline{u}_i$). The necessity of lines 8 and 14 are explained in Section 5.2.2.

Note that the critical point $u = u_3$ in Fig. 8 is a switch critical point. One real-world example of topological structure

4. For example, for the critical point u_3 in Fig. 8, there is a branch point which has four closest points on the boundary, i.e., on C^+, v_C, G_3 and G_4, respectively.

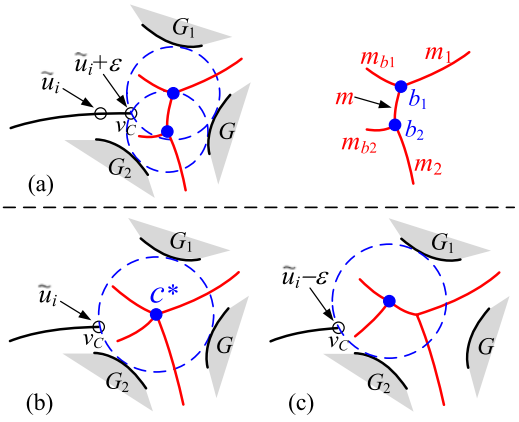


Fig. 10. Topological structure updating when a vanishing critical point $\widetilde{u}_i$ in $\Phi(v_C)$ is traversed. ε is a sufficiently small value.

updating with switch critical points in $\Phi(C^-)$ is shown in Fig. S3 in the supplemental material.

Algorithm 1. Update_Switch($\overline{u}_i$): Update the Symbolic Representation $\Phi(\Omega)$ of the Merged Medial Axis when a Switch Critical Point $\overline{u}_i$ is Traversed.

1: Let G be the generator that offers mating points to $(\overline{u}_i-\varepsilon, \overline{u}_i)$.
2: Let G' be the generator that offers mating points to $(\overline{u}_i, \overline{u}_i+\varepsilon)$.
3: If u_i is a delimiting point from C^+
4: Delete G' from $\Phi(C^+)$.
5: Delete C^+ from $\Phi(G')$.
6: Insert v_C into $\Phi(G)$ at the position before C^+.
7: Insert G into $\Phi(v_C)$ as the position after C^+.
8: Update the priority queue Q.
9: Else //u_i is a delimiting point from C^-
10: Delete G' from $\Phi(C^-)$.
11: Delete C^- from $\Phi(G')$.
12: Insert v_C into $\Phi(G)$ at the position after C^-.
13: Insert G into $\Phi(v_C)$ as the position before C^-.
14: Update the priority queue Q.

5.2.2 *Topological Structure Updating in* $\Phi(v_C)$

The critical point triggered by a generator leaving $\Phi(v_C)$ corresponds to the event that an MMI of v_C becomes zero-length. Suppose at $u=u_c$, an MMI on v_C attains zero length, implying that there is a medial curve m that exists (with non-zero length) in $(u_c, u_c+\varepsilon)$ for a sufficiently small $\varepsilon>0$, but reaches zero length at $u=u_c$.

Refer to Fig. 10. Let b_1 and b_2 be two branch points that delimit m in $M(\Omega(u_c+\varepsilon))$. Let m_{b1} and m_1 (resp. m_{b2} and m_2) be the other two medial curves incident at b_1 (resp. b_2), where m_{b1} and m_{b2} are formed by v_C and $\Phi(v_C)$. At $u=u_c$, these four medial curves meet at a common irregular branch point $c^*=c_1=c_2$. By Definition 10, u_c is a critical point at which a generator G is just about to leave $\Phi(v_C)$, and we refer to its type as *vanishing critical point*. One real-world example is shown in Fig. S4 in the supplemental material.

Vanishing critical points exist if there are at least three generators in $\Phi(v_C)$. Let $\Phi(v_C)=\{G_{m_1}, G_{m_2}, \ldots, G_{m_r}\}$, $r\geq 3$. Then for $i=2,3,\ldots,r-1$, each G_{m_i} contributes to a candidate vanishing critical point, which is recorded by its corresponding parameter value $\widetilde{u}_i$ on the separator C. Denote by $D_{i-1,i,i+1}$ the maximum disk in Ω determined by three generators $G_{m_{i-1}}$, G_{m_i} and $G_{m_{i+1}}$ (ref. dashed circle with center c^* in Fig. 10b). The value of the $\widetilde{u}_i$ is then computed as the intersection position of C and $\partial D_{i-1,i,i+1}$.

We sort all the candidate vanishing critical points $\{\widetilde{u}_2, \widetilde{u}_3, \ldots, \widetilde{u}_{r-1}\}$ in a priority queue Q. For line 7 in Algorithm 1, a new generator G_{m_0} enters $\Phi(v_C)$ and then a new candidate vanishing critical point $\widetilde{u}_1$ is computed as the intersection position of C and $\partial D_{0,1,2}$. Accordingly at line 8 in Algorithm 1, we add $\widetilde{u}_1$ into Q. At line 14 in Algorithm 1, a new $\widetilde{u}_r$ is added into Q, which is computed by the intersection of of C and $\partial D_{r-1,r,r+1}$, where $G_{m_{r+1}}$ is the new generator entering $\Phi(v_C)$ at line 13.

The maximal element in Q is a true vanishing critical point. Denote this maximal element by $\widetilde{u}_k$. When the critical point $\widetilde{u}_k$ is traversed, the generator G_k is leaving $\Phi(v_C)$ and we remove $\widetilde{u}_k$ from Q. Meanwhile, the candidate critical points $\widetilde{u}_{k-1}$ and $\widetilde{u}_{k+1}$ are obsolete, since they originally correspond to the triples $(G_{m_{k-2}}, G_{m_{k-1}}, G_{m_k})$ and $(G_{m_k}, G_{m_{k+1}}, G_{m_{k+2}})$ respectively. We compute the new values of $\widetilde{u}_{k-1}$ and $\widetilde{u}_{k+1}$ using the new triples $(G_{m_{k-2}}, G_{m_{k-1}}, G_{m_{k+1}})$ and $(G_{m_{k-1}}, G_{m_{k+1}}, G_{m_{k+2}})$ respectively. Finally, the positions of new $\widetilde{u}_{k-1}$ and $\widetilde{u}_{k+1}$ are updated in Q.

The local topological structure change by traversing a vanishing critical point is summarized in Algorithm 2: Update_Vanish($\widetilde{u}_k$).

Algorithm 2. Update_Vanish($\widetilde{u}_k$): Update the Symbolic Representation $\Phi(\Omega)$ of Medial Axis when a Vanishing Critical Point $\widetilde{u}_k$ is Traversed.

1: Let G_{m_k} be the generator in $\Phi(v_C)$ that corresponds to $\widetilde{u}_k$.
2: Delete G_{m_k} from $\Phi(v_C)$.
3: Delete v_C from $\Phi(G_{m_k})$.
4: Insert $G_{m_{k-1}}$ into $\Phi(G_{m_{k+1}})$ between G_{m_k} and v_C.
5: Insert $G_{m_{k+1}}$ into $\Phi(G_{m_{k+1}})$ between G_{m_k} and v_C.
6: Update the position of $\widetilde{u}_{k-1}$ in Q using the new value determined by $(G_{m_{k-2}}, G_{m_{k-1}}, G_{m_{k+1}})$.
7: Update the position of $\widetilde{u}_{k+1}$ in Q using the new value determined by $(G_{m_{k-1}}, G_{m_{k+1}}, G_{m_{k+2}})$.

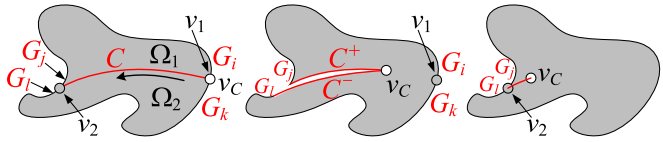


Fig. 11. The special initialization (middle) and termination (right) operations in the overall algorithm.

5.3 Overall Algorithm Implementation

Let two simple sub-shapes Ω_1 and Ω_2 share a boundary C as shown in Fig. 11 left, in which v_1 and v_2 are the intersection points of C and $\partial(\Omega_1\cup\Omega_2)$. To apply the proposed topology-oriented method, a special initialization operation is needed (ref. Fig. 11 middle), which is summarized in Algorithm 3: Initialization($\Phi(\Omega_1), \Phi(\Omega_2), C$).

During the shrinking process of C, when there is no critical point on C, the process is terminated. At this end, a special termination operation is also needed (ref. Fig. 11 right), which is summarized in Algorithm 4: Termination($\Phi(\Omega(v_C)), C$).

Algorithm 3. Initialization($\Phi(\Omega_1), \Phi(\Omega_2), C$)

1: Set $C^+ = C$ in $\Phi(\Omega_1)$.
2: Set $C^- = C$ in $\Phi(\Omega_2)$.
3: Let v_1 be an intersection point of C and $\partial(\Omega_1 \cup \Omega_2)$, and generate a new boundary generator v_C at the position v_1 such that C^+, v_C and C^- satisfy the predefined orientation in $\Omega_1 \cup \Omega_2$ (Fig. 11 left and middle).
4: Let G_i and G_j be the generators adjacent to v_1 (other than C) in Ω_1 and Ω_2 respectively.
5: Let S_{C^+} be the set of delimiting points of MMIs on C in Ω_1.
6: Let S_{C^-} be the set of delimiting points of MMIs on C in Ω_2.
7: If v_1 become convex after shape merging
8: Set $\Phi(v_C) = \{C^+, G_i, G_k, C^-\}$.
9: Insert v_C into $\Phi(C^+)$ at the back of G_i.
10: Insert v_C into $\Phi(C^-)$ in the front of G_k.
11: Else
12: Set $\Phi(v_C) = \{C^+, G_i, v_1, G_k, C^-\}$.
13: Insert v_C into $\Phi(C^+)$ at the back of v_1.
14: Insert v_C into $\Phi(C^-)$ in the front of v_1.

Algorithm 4. Termination($\Phi(\Omega(v_C), C)$)

1: Let v_2 be the intersection point of C and $\partial(\Omega_1 \cup \Omega_2)$ (Fig. 11 right).
2: Let G_j and G_l be the generators adjacent to v_2 (other than C) in Ω_1 and Ω_2 respectively.
3: Remove v_L and C^+ from $\Phi(G_j)$.
4: Remove C^- and v_L from $\Phi(G_l)$.
5: Remove C^+, C^-, v_L and $\Phi(C^+)$, $\Phi(C^-)$, $\Phi(v_L)$ from $\partial\Omega$ and $\Phi(\Omega)$, respectively.

Now we are ready to present the overall algorithm for merging two medial axes along a shared boundary, which is summarized in Algorithm 5: Merge_Medial_Axes ($\Phi(\Omega_1), \Phi(\Omega_2), C$). The following theorem summarizes its time complexity.

Theorem 1. *The time complexity of Algorithm 5 is $O(n \log n_v)$, where n is the number of generators in $\partial\Omega_1 \cup \partial\Omega_2$, n_v is the maximal number of elements that simultaneously exist in the priority queue Q during the merging process, and n_v is strictly smaller than n. Furthermore, the medial axis $M(\Omega)$ can be constructed in $O(K\ n \log n_v)$ time, where K is the depth of recursion in a divide and conquer algorithm.*

We note that in all our experiments for two progressive medial axis computations (Section 6), n_v behaves as a small constant (ref. Figs. 17 and 20 for the distributions of n_v in real-world polygonal shapes) and thus Algorithm 5 performs as a linear algorithm. If Ω is a polygonal domain, K can be optimally $O(\log n)$ and $M(\Omega)$ are constructed in $O(n \log n \log n_v)$ time.

So far we have assumed that both sub-shapes Ω_1 and Ω_2 are simple. However, our method does not need such a restriction. Without loss of generality, assume Ω_1 is not simple, i.e., there are $m \geq 1$ holes in it. We apply the augmented domain technique in [3] to convert Ω_1 into a simple shape.

Refer to Fig. 12. For each hole h_i inside Ω_1, we randomly pick up a generator G from ∂h_i and a generator G' from $\Phi(G)$, such that $G' \notin \partial h_i$, and $G' \neq C$. Then we randomly select a medial point x in the medial curve $m(G, G')$. Denote

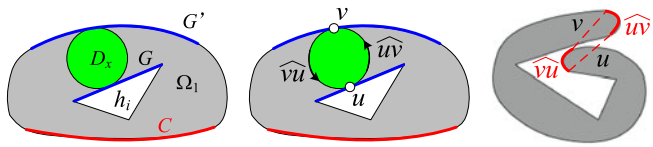


Fig. 12. An augmented domain of a non-simple shape Ω_1.

by D_x the maximal disk in Ω_1 centered at x. Let the mating pair of points of x be $\{u, v\}$. Points u and v partition ∂D_x into two arcs $\widehat{uv}$ and $\widehat{vu}$. Then the augmented domain is defined as $\Omega_1' = \Omega_1^0 \cup D_x^1 \cup D_x^2$, where $\Omega_1^0 = \{(p, 0) | p \in \Omega_1 \setminus D_x\}$, $D_x^1 = \{(p, 1) | p \in D_x\}$ and $D_x^2 = \{(p, 2) | p \in D_x\}$. Two points (p, i) and (q, j) are connected in Ω_1' if one of the following conditions is satisfied:

1) $i = j$ and the line segment $\overline{pq}$ avoids ∂D_x.
2) $i, j = \{0, 1\}$ and $\overline{pq}$ intersects the arc $\widehat{uv}$.
3) $i, j = \{0, 2\}$ and $\overline{pq}$ intersects the arc $\widehat{vu}$.

Algorithm 5. Merge_Medial_Axes($\Phi(\Omega_1), \Phi(\Omega_2), C$): The Input are Symbolic Representations $\Phi(\Omega_1)$ and $\Phi(\Omega_2)$, and a Shared Boundary C.

1: Initialization($\Phi(\Omega_1), \Phi(\Omega_2), C$); see Algorithm 3.
2: Merge S_{C^+} and S_{C^-} into an array S, in which elements are in descending order.
3: Initialize a priority queue Q as an empty set.
4: Set $i = 0$.
5: While ($i < |S|$ or Q is not empty)
// $|S|$ is the number of elements in S.
6: Extract the vanishing critical point $\widetilde{u}$ with the maximal value from Q.
7: If ($i == |S|$)
8: $S[i] = 0$;
9: If (element $\widetilde{u} \neq NULL$ and its value $\widetilde{u} > S[i]$)
10: Update_Vanish($\widetilde{u}$); see Algorithm 2.
11: Else
12: Update_Switch($S[i]$); see Algorithm 1.
13: $i++$;
14: Termination($\Phi(\Omega(v_C)), C$); see Algorithm 4.

One D_x removes one hole in Ω_1. We recursively find a D_x and remove a hole at each iteration until Ω_1 becomes simple.

Finally, if the separator C consists of several generators as long as their combination is still a simple open curve (e.g., a polyline with several line segments), we iteratively set C to be each of the generators and shrink them one by one.

6 Experiments

Algorithm 5 is general for planar shapes defined in Definition 1. We implemented Algorithm 5 for shapes whose generators consist of line segments and reflex vertices, and propose two progressive medial axes of polygonal shapes in this section. One is region-growing progressive medial axes (Section 6.1) and the other is boundary-evolved progressive medial axes (Section 6.2).

In addition to providing an important progressive representation of medial axes, our method has two more merits. First, our method is topology-oriented and thus is robust. Secondly, our method is fast for large-scale shapes with hundreds of thousands of generators. To demonstrate these

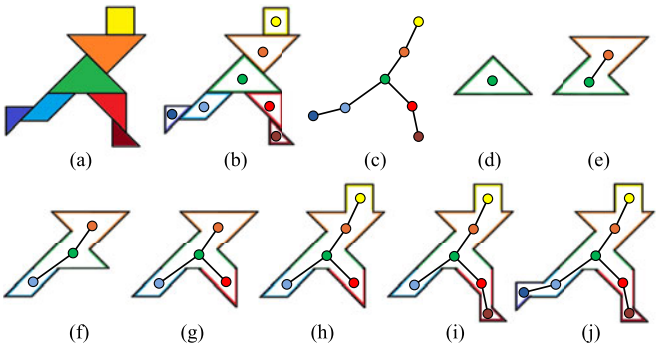


Fig. 13. (a) A schematic diagram of shape decomposition. (b) Each part corresponds to a graph node. (c) Build an adjacency graph by connecting two nodes by an edge if the corresponding two parts share some boundary. (d)-(j) A progressive region growing is performed by breadth-first search of the adjacency graph rooted at the green node.

merits, in Sections 6.1-6.3, we compare our method with state-of-the-art implementation of two robust geometric algorithms:

- Exact computation in CGAL using `CORE::Expr` as the predicate kernel [9].
- A floating-point filter technique in CGAL [7], [24], which is implemented in the class `2D Segment Delaunay Graphs` using `SegmentDelaunay GraphTraits_2` as the template parameter.[5]

In Section 6.4, we compare our method with other representative medial axis computation methods.

6.1 Region-Growing Progressive Medial Axis

Shape decomposition methods partition a complex planar shape into a set of simpler parts, and one schematic illustration is shown in Fig. 13a. We construct a region-growing progressive shape by building an adjacency graph of the decomposed shape, in which each node represents a part (Fig. 13b) and two nodes are connected by an edge if the corresponding two parts share some boundary (Fig. 13c). Then starting with any node (e.g., a node whose related part has the maximal ordinate coordinate or has the largest area), the shape region is progressively grown by breadth-first search of the adjacency graph (Figs. 13d-13j)). A real-world example is shown in Fig. 14.

We compute progressive medial axes of the region-growing progressive shape by iteratively merging one part into the progressive shape. Note that our progressive medial axis works for any segmentation-based shape decomposition ([16] for an example). In this section, for a clear illustration, we use the polygon triangulation algorithm [25] to decompose a complex shape into a set of triangles and use this triangulation to run all the results summarized in Table 1. Fig. 15 shows an example of this kind of progressive medial axes. Seven complex shapes shown in Fig. 16 are tested and their progressive medial axes are illustrated in the supplemental material (Figs. S8-S14).

During the region-growing, at each iteration, one triangle is merged into the progressive shape. For each progressive shape, at iteration i, we record the value n_v^i which is the

5. We use the default setting `bool fpFilterFlag = true` in CGAL/CORE/CoreDefs.cpp.

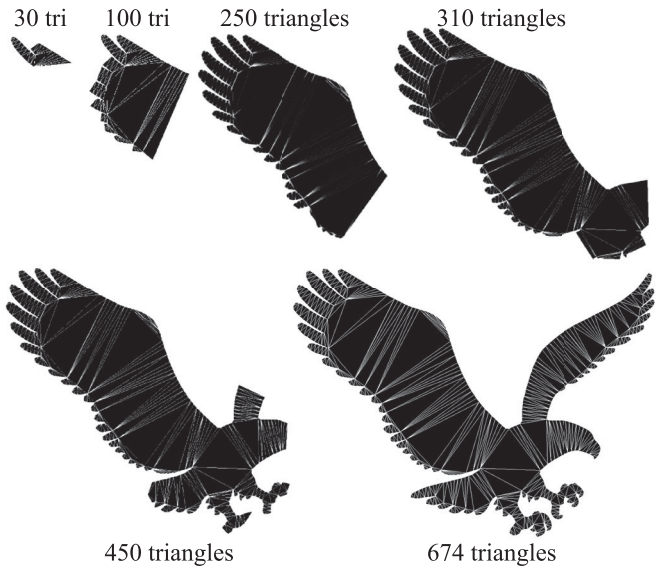


Fig. 14. A region-growing progressive shape starting at a triangle in which a vertex has the maximal ordinate coordinate. For a clear illustration of the shape as well as its medial axis shown in Fig. 15, a simplified shape (674 triangles) is used here. A full-resolution shape (10,833 triangles) is shown in Fig. 16.

number of elements that exists in the priority queue in Algorithm 5. Then the value n_v stated in Theorem 1 is $n_v = \max_i\{n_v^i | i, 1, 2, \ldots\}$. Fig. 17 shows the value n_v^i at every iteration and the results show that n_v behaves as a small constant with real-world complex shapes. Accordingly, our topology-oriented method (Algorithm 5) behaves as a linear algorithm. Since any triangulation of a simple polygon P with n vertices consists of $n-2$ triangles, a region-growing progressive medial axis of P consists of $n-2$ medial axes. By Theorem 1, our method computes these $n-2$ medial axes in $O(n^2)$ time. One example of the run time for the progressive medial axis (including 39,551 medial axes) of the Coral shape is illustrated in Fig. S5 in the supplemental material.

More results of running time are summarized in Table 1, showing that in addition to providing an on-the-fly progressive medial axis representation, our method is faster than the two robust implementations (exact computation and floating-point filter) in CGAL. In particular, for large-scale polygonal shapes with vertex number larger than 10 K (not including the shrimp data in Table 1), our method is 4 to 24 times faster than the floating-point filter technique implemented in CGAL.

Compared with existing medial axis computation methods [1], our region-growing progressive medial axis computation has the following merits:

- The progressive medial axis always starts with a triangle whose medial axis is readily obtained. Then at each iteration, a triangle is merged into the progressive shape locally in a topology-oriented way. Therefore, our method can compute the medial axis of complex shape quickly and robustly.
- It is well known that a small change in a shape may lead to a significant change of its medial axis. Our method offers a robust implementation of updating medial axes for shapes with small changes.

TABLE 1
The Comparison of CGAL and Our Topology-Oriented Method in the Application of Region-Growing Progressive Medial Axes of Polygonal Shape Data Shown in Fig. 16

Shape	Vertex number	Running time (seconds)				
		CGAL		Our topology-oriented method		
		Exact computation	Floating-point filtering	Triangulation	Region-growing progressive medial axis	Total time
Shrimp	681	96.7	0.468	0.006	0.273	0.279
Eagle	10,833	50,560.4	88.126	0.307	3.743	4.050
Dragon	16,865	48,389.1	73.172	0.402	6.243	6.645
Phoenix	17,153	74,556.3	157.884	0.535	6.236	6.771
Bamboo	19,921	60,285.0	120.479	0.429	8.036	8.465
Tree	33,457	48,289.5	73.282	0.498	17.687	18.185
Coral	39,553	114,161.0	138.112	0.893	24.678	25.571

All polygonal shapes are normalized by finding the minimum-area bounding box and rescaling the diagonal length of the bounding box to be one. The running time is measured on a PC (Intel(R) Core(TM) I7920 CPU 2.67GHz) running Windows 7.

6.2 Boundary-Evolved Progressive Medial Axis

The evolution of closed smooth planar curves by diffusion equations [26] and the evolution of closed polygonal curves by polyline simplification [15] have been well studied. We use the method in [15] to construct a boundary evolution of polygonal shape: at every evolution step, two adjacent boundary edges are replaced by a single line segment joining the same endpoints. The substitution is recursively performed using a measure on the significance of shape change. An example is shown in Fig. 18.

Given an original, detailed planar shape Ω^0, boundary evolution generates a set of simplified shapes $\{\Omega^1, \Omega^2, \ldots, \Omega^m\}$. To compute the medial axis $M(\Omega^0)$, we start from the most simplified version $M(\Omega^m)$ and refine the shape by applying the simplification in reverse order. There are two types of refinements:

- Two adjacent boundary edges are added outside the current shape Ω^i; i.e., a new triangle Δ_{new} is added to Ω^i. We apply the proposed method to merge the medial axes of Δ_{new} and Ω^i along the shared boundary; an example is shown in Fig. 19.
- Two adjacent boundary edges are added inside the current shape Ω^j (see evolution step 13 in Fig. 18 for an example). In this case, the medial curves in $M(\Omega^j)$ intersecting two added boundary generators need to be updated locally. We apply the topology-oriented incremental Voronoi diagram method [13] for this local updating.

The computation of progressive medial axes in boundary evolution shares the same merit in region-growing progressive medial axis computation, that is, it recursively updates the progressive shape by adding/subtracting triangles locally in a topology-oriented way. Thus it can compute the medial axis of complex shape quickly and robustly.

We tested the polygonal shapes of six continents extracted from the topographical data of U.S. Geological Survey. One progressive evolution of both the Africa model and its medial axis is shown in Fig. 1. The others are illustrated in the supplemental material (Figs. S15-S20). Fig. 20 shows the value n_v^i at every iteration for each model. The experimental

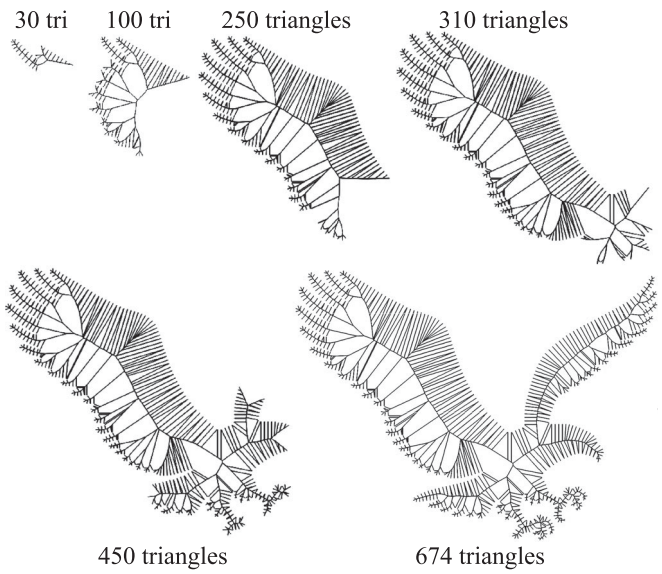


Fig. 15. The progressive medial axes of the region-growing progressive shape shown in Fig. 14.

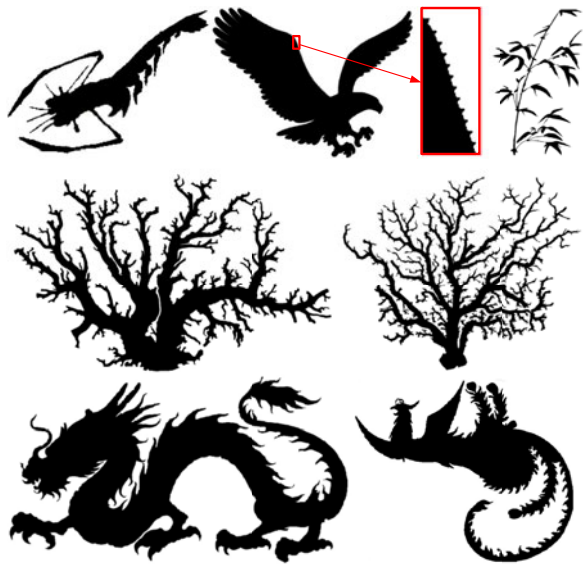


Fig. 16. Seven complex shapes, shrimp, eagle, bamboo, coral, tree, dragon, and phoenix, are tested. Results are summarized in Table 1.

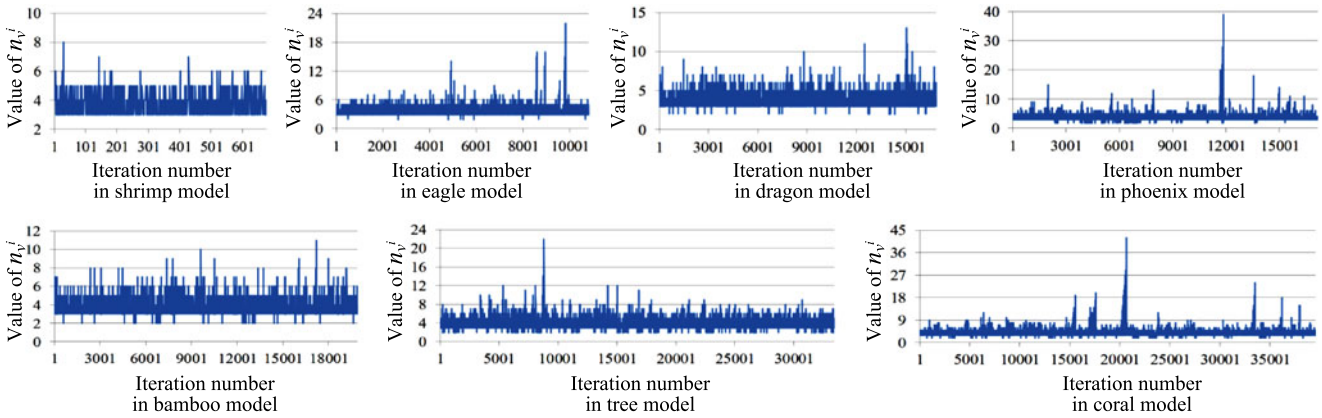


Fig. 17. In the refinement process of region-growing progressive medial axes for each model in Table 1, the value n_v in Theorem 1, which is the maximal number of elements that simultaneously exist in the priority queue Q in Algorithm 5, behaves as a small constant.

results show that the value of n_v stated in Theorem 1 behaves as a small constant, which is consistent with the observation in Fig. 17. In this progressive representation, our method computes $O(n)$ medial axes in $O(n^2)$ time. One example of the run time for the progressive medial axis of the Asia model is illustrated in Fig. S6 in the supplemental material.

The statistic data of the performance of our boundary-evolved progressive medial axis as well as the comparison with the floating-point filter in CGAL are summarized in Table 2. For all the large-scale polygonal shapes in Table 2, the exact computation in CGAL is unable to output results after running one week. So we only present the time of CGAL with a floating-point filter. The program of CGAL FPF even crashed when handling the data of North American model and Oceania model. These results demonstrate that in addition to providing an on-the-fly progressive medial axis representation, our boundary-evolved progressive medial axis computation is robust and faster than two robust geometric algorithms in CGAL by at least an order of magnitude.

6.3 Accuracy Evaluation

Our method is primarily described in terms of combinatorial and topological computation, and relies on the numerical computation[6] moderately. To evaluate its numerical accuracy, we compare our method with the exact computation in CGAL.

The exact computation in CGAL can represent numbers of arbitrary precision. Although it computes exact medial axes, these results cannot be directly used for downstream applications, if these applications use fixed-precision numbers only. Accordingly, in the class `CORE::Expr` of CGAL, an approximation function `approx` is provided to convert numbers of arbitrary precision into the IEEE double format. `approx` computes an approximation with a combined precision $(relPrec, absPrec)$, i.e., if e is the exact value and ee is the approximate value, then $|e - ee| \leq 2^{-absPrec}$ or $|e - ee| \leq 2^{-relPrec}|e|$. The default setting in `CORE::Expr` is $relPrec = 53$ and $absPrec = 1{,}024$.

We represent the output medial axes in the IEEE double format. For each branch point x in $M(\Omega)$ of a polygonal shape Ω, due to numerical errors, let $r_{\min} = \min_i\{\|x - y_i\|_2 : y_i \in \Lambda(x)\}$ and $r_{\max} = \max_j\{\|x - y_j\|_2 : y_j \in \Lambda(x)\}$. Then we use $E(x) = 1 - \frac{r_{\min}}{r_{\max}}$ as a normalized error measure at x. We sort the values $E(x)$ of all branch points in decreasing order and plot the error curves of our method and the exact computation in CGAL. The error curves of seven shapes in Fig. 16 are illustrated in Fig. S7 in the supplemental material. The means and standard deviations of $E(x)$ values in these seven shapes are summarized in Table 3. These results

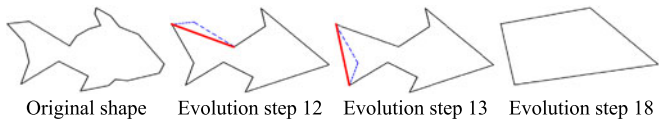


Fig. 18. Boundary evolution of a polygonal fish shape. Starting from an original shape (leftmost), at every evolution step, two adjacent boundary edges (shown in dashed blue color) are replaced by a single line segment (shown in red) joining the same endpoints.

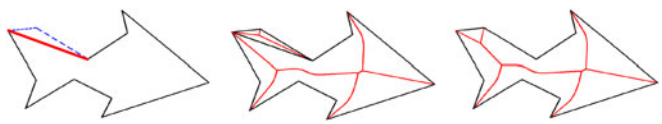


Fig. 19. Two adjacent boundary edges are added outside of the current shape (evolution step 12 in Fig. 18) and medial axes are merged along shared boundary.

TABLE 2
The Comparison of CGAL with a Floating-Point Filter and our Boundary-Evolved Progressive Medial Axis Method, on Large-Scale Polygonal Shape Data Extracted from U.S. Geological Survey

Shape	Vertex number	Running time (minutes)	
		CGAL FPF	Our method
Africa	204,545	325.1	14.8 (1.7+13.1)
Asia	115,841	60.1	5.2 (1.0+4.2)
Europe	124,097	58.5	5.6 (1.1+4.5)
North America	274,241	crashed	23.2 (2.4+20.8)
South America	190,817	254.4	12.3 (1.5+10.8)
Oceania	147,713	crashed	8.8 (1.3+7.5)

The CGAL exact computation time is not included because it simply takes too much time compared to the others. The running time of our method consists of two parts $(a + b)$: time a for shape evolution generation and time b for progressive medial axis generation. All polygonal shapes are normalized by rescaling the diagonal length of the bounding box to be one. The running time is measured on a PC (Intel(R) Core(TM) I7920 CPU 2.67GHz) running Windows 7.

6. The numerical computation in Algorithm 5 is based on the code in [27].

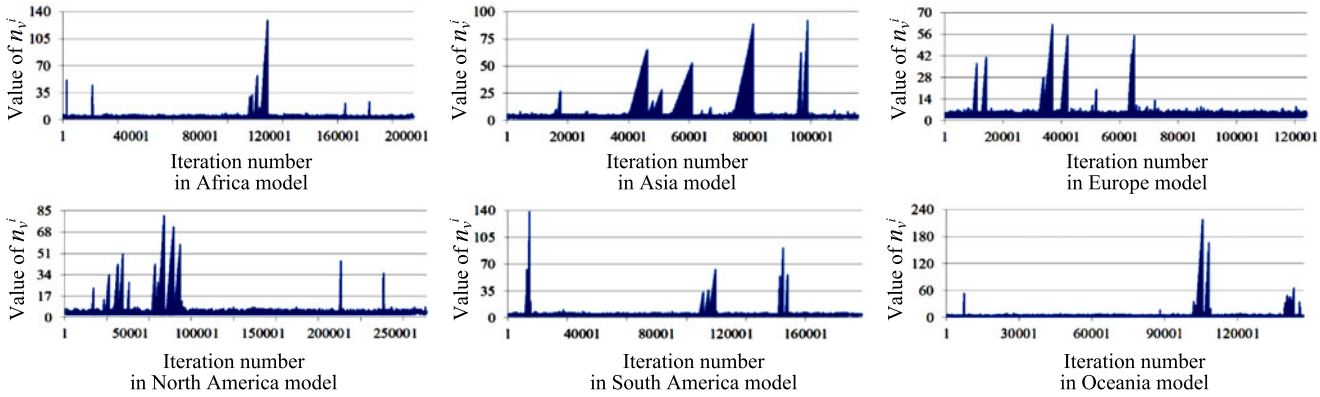


Fig. 20. In the refinement process of boundary-evolved progressive medial axes for each model in Table 2, the value n_v in Theorem 1, which is the maximal number of elements that simultaneously exist in the priority queue Q in Algorithm 5, behaves as a small constant.

TABLE 3
The Means and Standard Deviations (SD) of Normalized Errors $E(x)$ for All Branch Points in Seven Shapes (Fig. 16), by Our Method and the Exact Computation in CGAL

	$E(x)$	Shrimp	Eagle	Dragon	Phoenix	Bamboo	Tree	Coral
Our	Mean	2.2×10^{-10}	8.1×10^{-6}	1.3×10^{-6}	2.3×10^{-7}	2.5×10^{-12}	4.3×10^{-12}	1.0×10^{-5}
Method	SD	1.1×10^{-8}	1.1×10^{-3}	1.8×10^{-4}	2.6×10^{-5}	1.6×10^{-11}	2.4×10^{-11}	2.0×10^{-3}
CGAL	Mean	4.5×10^{-7}	6.2×10^{-7}	5.5×10^{-7}	9.8×10^{-7}	5.6×10^{-7}	8.1×10^{-7}	1.5×10^{-6}
EXACT	SD	2.1×10^{-6}	8.2×10^{-6}	5.0×10^{-6}	6.1×10^{-6}	6.8×10^{-6}	4.9×10^{-6}	1.9×10^{-5}

The error curves of $E(x)$ in seven shapes are illustrated in Fig. S7 in the supplemental material.

show that our method has comparable numerical accuracy with the exact computation in CGAL; that is, our method is better in the data of Shrimp, Phoenix, Bamboo and Tree, while the CGAL exact computation is better in the data of Eagle, Dragon and Coral.

6.4 Comparison with Representative Medial Axis Computation Methods

Merging two sub-shapes into one is a necessary step in a divide-and-conquer algorithm. In this section, we compare our method with three representative divide-and-conquer methods.

An early classic divide-and-conquer algorithm was proposed in [6]. This method partitions a shape boundary into two halves H_1 and H_2, and build the Voronoi diagrams $VD(H_1)$ and $VD(H_2)$ respectively. An example in [6] is shown in Fig. 21 in which the two halves of generators are $H_1 = \{g_1, g_2, \ldots, g_{10}\}$ and $H_2 = \{g_{11}, g_{12}, \ldots, g_{21}\}$. The merging step starts at a starting bisector $B(g_{10}, g_{11})$ of generators g_{10} and g_{11}, and ends at a terminating bisector $B(g_1, g_{21})$ (Fig. 21c). A numerical tracing is used in [6] for this merging. For example, given the merging bisector $B(g_9, g_{13})$, the algorithm scans the edge of Voronoi polygon $V(g_9)$ in $VD(H_1)$ in the CCW direction to find the edge which intersects $B(g_9, g_{13})$. A similar scan is performed in Voronoi polygon $V(g_{13})$ in $VD(H_2)$ in the CW direction. Then the numerical values are compared to determine which edge intersects $B(g_9, g_{13})$ first. If the two intersection points are very close as shown in this example, the local tracing process may be error-prone due to numerical imprecision. As a comparison, our method is robust by adding triangles one-by-one and updating merged medial axes with topological description of mating generator lists. Both region-growing and boundary-evolved progressive medial axes by applying our method to the same example are shown in Fig. 22.

Targeting on a robust implementation, a representative divide-and-conquer algorithm was proposed in [5] based on a domain decomposition lemma [18]. Refer to Fig. 23. Let b be a branch point of the medial axis $M(\Omega)$. Denote by $D(b)$

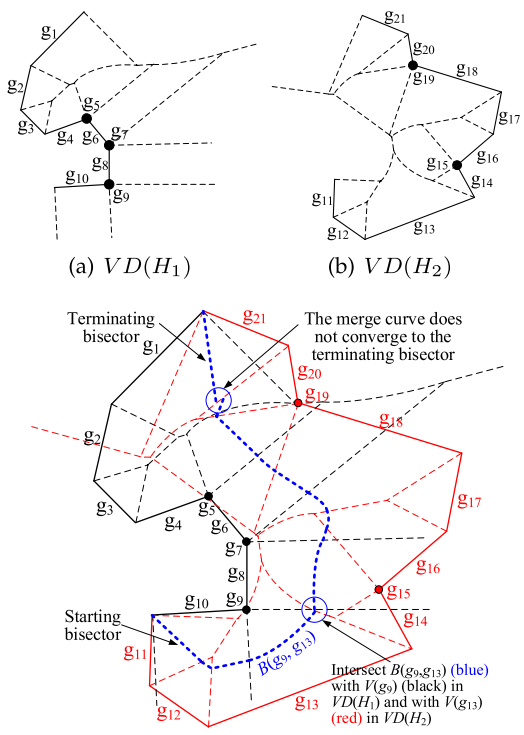


Fig. 21. An example in a classic divide-and-conquer algorithm in [6].

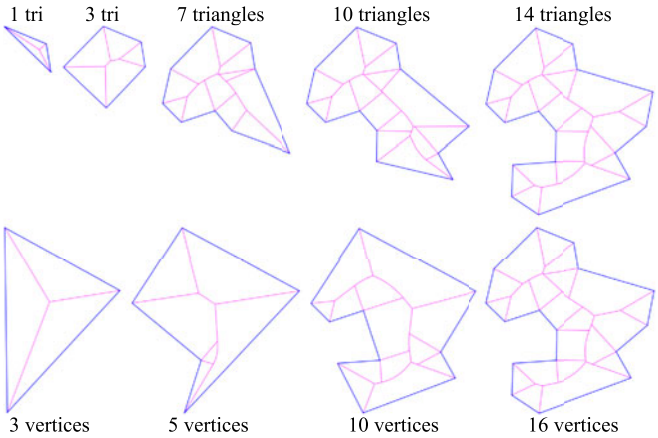


Fig. 22. Robust medial axes computation using our method for the same example in Fig. 21. Top row: region-growing progressive medial axes. Bottom row: boundary-evolved progressive medial axes.

the maximal disk centered at b in Ω. If $\Omega \setminus D(b)$ has $k > 1$ components $\{A_1, A_2, \ldots, A_k\}$, the domain decomposition lemma [18] says that $M(\Omega) = \bigcup_{i=1}^{k} M(\Omega_i)$, where $\Omega_i = A_i \cup D(b)$. The algorithm in [5] decomposes the shape by numerically locating all branch points and terminal points[7] by a steepest descent method and stores their connectivity information in a novel tree structure.

The domain decomposition lemma was further used in a state-of-the-art divide-and-conquer algorithm [3] in which a multiple objects' domain is recursively split. The advantages of the algorithm in [3] are that (1) the Voronoi diagram of the multiple objects' domain can be computed stably by means of applying a medial axis algorithm in a simply connected domain, and (2) the necessary combinatorial structure of the medial axis can be constructed without numerically computing the trimmed bisectors explicitly.

Our proposed method computes the medial axis in a similar fashion of [3], [5], but with two improved characteristics. The first is that our method is topology-oriented so that it is very robust. The second is that the previous methods [3], [5] decompose the shape using their own specific rules, while our method works for arbitrary shape decomposition as long as the sub-shapes share a common boundary. One example is shown in Fig. 24, in which two sub-shapes Ω_1 and Ω_2 of north and south America are merged into the whole America continent shape $\Omega = \Omega_1 \cup \Omega_2$. The shapes Ω_1, Ω_2 and Ω have respectively 274,241, 190,817 and 465,046 boundary generators. If the medial axes $M(\Omega_1)$ and $M(\Omega_2)$ are available, our method can efficiently compute $M(\Omega)$ by local updating in less than 1 second. Note that all the previous divide-and-conquer methods [3], [5], [6] fail to merge medial axes in this case. Note also that other medial axis computation methods such as the randomized incremental

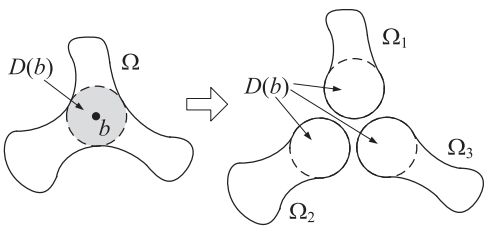


Fig. 23. The domain decomposition scheme used in [5].

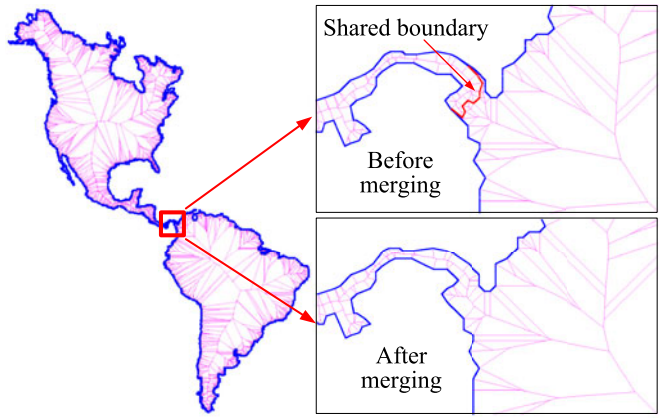


Fig. 24. Merge north and south America into the whole continent shape. Due to limited resolution, only 2,595 boundary generators are shown in this figure. The original whole America shape has 465,046 boundary generators. If the medial axes of north and south America shapes are available, our method take less than 1 second to obtain the merged medial axis.

algorithm [24] in CGAL cannot merge medial axes in this case either.

Our topology-oriented method is also related to the VRONI [28], which is an industrial-strength implementation for computing Voronoi diagrams of points and line segments, and its ArcVRONI extension to circular arcs [29]. However, neither VRONI nor ArcVRONI can be used to compute a progressive representation of medial axes.

7 Conclusions

In this paper, we propose to compute progressive medial axes of large-scale planar shapes in a fast and robust way. The key ingredient is a topology-oriented method to merging medial axes of two planar shapes along a shared boundary. Several topological properties are proved, which are used to characterize the structural changes in medial axes using two types of critical points. Our method can work with any third-party shape decomposition methods, in which two (region-growing and boundary-evolved) progressive medial axes representations are implemented. The experimental results show that in addition to providing an on-the-fly progressive representation, our topology-oriented method is faster than two robust implementations (exact computation and floating-point filtering) in CGAL.

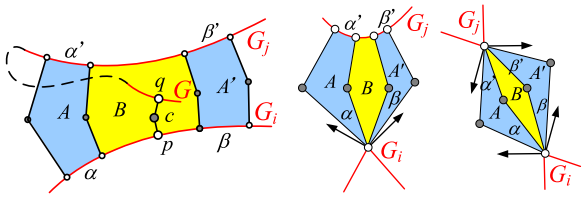
Future work includes the extension to medial axis computation for 3D solid shapes. If the shape is represented by a polyhedron, a straight-forward way is to use a tetrahedronization for building a progressive representation. In this case, the types of critical points and their distribution on the separating plane deserve to be investigated carefully.

Appendix

Lemma 1. *Suppose that Ω is simple. On a mating pair of generators $\{G, G'\}$, there exists exactly one mating pair of MMIs.*

Proof. We prove the case when the two mating generators are edges, say G_i and G_j (Fig. 25a), as the situation involving reflex vertices is similar (Fig. 25b). Refer to Fig. 25a. If Lemma 1 does not hold, there are at least two mating pairs

7. A point $x \in \overline{M}(\Omega)$ is a terminal point if $|\Lambda(x)| = 1$.



(a) The edge/edge case (b) The vertex/edge and vertex/vertex cases

Fig. 25. Proof of Lemma 1.

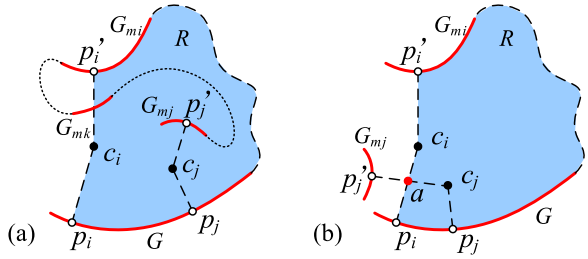


Fig. 26. Proof of Lemma 4.

of MMIs $\{\alpha, \alpha'\}$ and $\{\beta, \beta'\}$ existing on the mating pair of generators $\{G_i, G_j\}$, in which $\alpha \bigcap \beta \neq \emptyset$ and $\alpha' \bigcap \beta' \neq \emptyset$ cannot be held at the same time, otherwise $\{\alpha \bigcup \beta, \alpha' \bigcup \beta'\}$ would be a single mating pair. Without loss of generality, we assume $\alpha \bigcap \beta = \emptyset$. Together with normal vectors at endpoints, the four intervals $\alpha, \alpha', \beta, \beta'$ introduce three regions, shown as A, A', B in Fig. 25a. By the definition of medial axis and mating relation, the interior of regions A and A' are clear of any boundary generators. Since $\alpha \bigcap \beta = \emptyset$, we consider a point p on G_i, which satisfies $p \notin \alpha$, $p \notin \beta$ and $mate(p) \notin G_j$. Let G be the generator that the mating point $q = mate(p)$ belongs to. Both the mating point q and its medial point c must be strictly inside region B, otherwise at least one of the two conditions $dist(c, p) = dist(c, \partial\Omega)$ and $dist(c, q) = dist(c, \partial\Omega)$ would be violated. This indicates that the generator G must lie inside region B. Since Ω is simple, G is connected to other generators in $\partial\Omega$. Therefore, there is a chain of generators that connect G to G_j (or G_i) which has to go through region A (or A'). This contradicts the assumption that the interior of both A and A' are clear of generators. □

Lemma 2. *$G_i \in \Phi(G_j)$ if and only if $G_j \in \Phi(G_i)$.*

Proof. If $G_i \in \Phi(G_j)$, then the pair $\{G_i, G_j\}$ contributes to a medial curve $m_{ij} = m(G_i, G_j)$. Therefore $G_j \in \Phi(G_i)$ and vice versa. □

Lemma 3. *Let G be a generator in a simple shape Ω. All the generators in $\Phi(G)$ are distinct.*

Proof. If two generators $G_i, G_j \in \Phi(G)$, $i \neq j$, are the same, i.e., $G_i = G_j$, then between G_i and G there are two distinct mating pairs of MMIs, i.e., $\{\alpha_i, mate(\alpha_i)\}$ and $\{\alpha_j, mate(\alpha_j)\}$, $i + 1 < j$. A contradiction to Lemma 1. □

Lemma 4. *(Inverse order preservation lemma.) Let the generators of a simple shape Ω be ordered using the in-front relation with respect to a generator G, i.e., $G_1 \prec_G G_2 \prec_G \cdots \prec_G G_n$. In the mating generator list $\Phi(G) = \{G_{m_1}, G_{m_2}, \ldots, G_{m_r}\}$, $G_{m_j} \prec_G G_{m_i}$, $i, j \leq r \leq n$, if and only if $j > i$.*

Proof. We prove the case that G and the boundary generators in $\Phi(G)$ are all edges, as the situation involving reflex vertices is similar. Refer to Fig. 26. Let $p_i = G(u_i)$ and $p_j = G(u_j)$ be two points on G, $0 < u_i < u_j < 1$, whose mating points $p_i' = mate(p_i)$ and $p_j' = mate(p_j)$ lie on generators G_{m_i} and G_{m_j}, respectively.

For a sufficient condition, given $u_i < u_j$, we need to prove $G_{m_j} \prec_G G_{m_i}$. Suppose it is not true and $G_{m_i} \prec_G G_{m_j}$. Let $c_i \in M(\Omega)$ be the medial point of mating pair $\{p_i, p_i'\}$ and R be the region enclosed by the ordered generators $\{G, G_{m_1}, \ldots, G_{m_i}\}$ and the two line segments $\overline{c_i p_i}$ and $\overline{c_i p_i'}$. As illustrated in Fig. 26a, the generator G_{m_j} cannot lie inside region R, since if so the simple connectedness of Ω would require some generator G_{m_k} between G_{m_i} and G_{m_j} to intersect one of two line segments $\overline{c_i p_i}$ and $\overline{c_i p_i'}$. This however would eliminate c_i from $M(\Omega)$, a contradiction. If G_{m_j} is outside region R, on the other hand, one of two line segments $\overline{c_j p_j}$ and $\overline{c_j p_j'}$ would intersect one of $\overline{c_i p_i}$ and $\overline{c_i p_i'}$, where c_j is the medial point of $\{p_j, p_j'\}$. There are a limited number of intersection configurations and Fig. 26b shows one of them in which $\overline{c_i p_i}$ intersects $\overline{c_j p_j'}$ at a point a. If $dist(a, p_i) \geq dist(a, p_j')$, c_i cannot be a medial point since $dist(c_i, p_j') < dist(c_i, a) + dist(a, p_j') \leq dist(c_i, p_i)$. Conversely, if $dist(a, p_i) < dist(a, p_j')$, c_j cannot be a medial point. All the remaining configurations can be processed with the same arguments.

By Lemma 1, since each G_{m_i} corresponds to only one MMI α_i on G, the necessary condition is readily obtained. This completes the proof. □

Lemma 5. *Let $\Pi(u)$ represent the union of all mating relations in the shape $\Omega(u)$. Suppose $0 < u_2 < u_1 < 1$. A mating pair of points $\{p, q\} \in \Pi(u_2)$ does not belong to $\Pi(u_1)$, if and only if there exist two distinct parameters $u_p, u_q \in (u_2, u_1]$ such that $\{p, C(u_p)\}$ and $\{q, C(u_q)\}$ are two mating pairs of points in $\Pi(u_1)$.*

Proof. First, if $\{p, q\}$ is a mating pair of points in $\Pi(u_1)$, $p, q \notin C(u)|_{u \in (0, u_1]}$, then $\{p, q\}$ will still be in $\Pi(u_2)$ since the separator C is always shrunk. Next, if $\{p, q\} \in \Pi(u_2)$ and $\{p, q\} \notin \Pi(u_1)$, then at least one of p and q forms a pair of mating points in $\Pi(u_1)$ with a point in $C(u)|_{u \in (u_2, u_1]}$. Assume $\{q, C(u_q)\} \in \Pi(u_1)$, $u_2 < u_q \leq u_1$. If Lemma 5 is not true, then there exists $\{p, p'\} \in \Pi(u_1)$, $p' \notin C(u)|_{u \in (u_2, u_1]}$. Refer to Fig. 27. Let c_1 be the medial point of $\{p, q\}$ in $\Pi(u_2)$ and c_2, c_3 be medial points of $\{p, p'\}$, $\{q, C(u_q)\}$ in $\Pi(u_1)$, respectively. We have $dist(c_1, p) \leq dist(c_2, p)$, otherwise $\{p, q\}$ cannot be in $\Pi(u_2)$. Then we have $dist(c_2, C(u_q)) < dist(c_2, c_1) + dist(c_1, C(u_q)) < dist(c_2, c_1) + dist(c_1, c_3) + dist(c_3, C(u_q)) = dist(c_2, c_1) + dist(c_1, q) = dist(c_2, p)$. A contradiction to the assumption $\{p, p'\} \in \Pi(u_1)$. □

Theorem 1. *The time complexity of Algorithm 5 is $O(n \log n_v)$, where n is the number of generators in $\partial\Omega_1 \cup \partial\Omega_2$, n_v is the maximal number of elements that simultaneously exist in the priority queue Q during the merging process, and n_v is strictly*

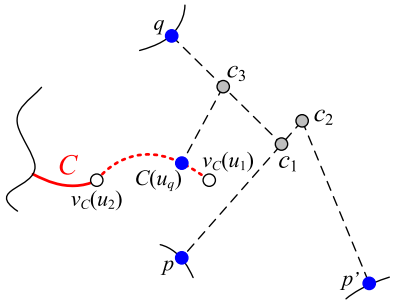


Fig. 27. Proof of Lemma 5.

smaller than n. Furthermore, the medial axis $M(\Omega)$ can be constructed in $O(Kn \log n_v)$ time, where K is the depth of recursion in a divide and conquer algorithm.

Proof. Note that (1) for each vanishing critical point being traversed, the number of vanishing critical points is reduced by one and the mating generator lists of four generators are updated in $O(1)$ time, and (2) for each switch critical point being traversed, the number of switch critical points is reduced by one, and the number of vanishing critical points is increased by one, and (3) the mating generator lists of four generators are updated in $O(1)$ time. It is clear that there are $O(n)$ switch critical points and it takes $O(n)$ time to process them. We use a priority queue to process the vanishing critical points and its time complexity is $n_v \log n_v$, $n_v < n$. Finally note that both Algorithms 3 and 4 take constant time. □

ACKNOWLEDGMENTS

The topographical data used in Table 2, Figs. 1 and 24 is courtesy of U.S. Geological Survey. The authors also thank the CGAL Open Source Project (https://www.cgal.org/) to make the CGAL software public available. This work was supported by the Natural Science Foundation of China (61322206, 61521002, 61432003) and the NRF grant funded by the Korea government (2012R1A2A1A05026395).

REFERENCES

[1] K. Siddiqi and S. Pizer, *Medial Representations: Mathematics, Algorithms and Applications*, New York, NY, USA: Springer, 2008.
[2] O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler, and M. Rabl, "Medial axis computation for planar free-form shapes," *Comput.-Aided Des.*, vol. 41, no. 5, pp. 339–349, 2009.
[3] O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler, E. Pilgerstorfer, and M. Rabl, "Divide-and-conquer for Voronoi diagrams revisited," in *Proc. 25th Annu. Symp. Comput. Geom.*, 2009, pp. 189–197.
[4] Y.-J. Liu, "Semi-continuity of skeletons in two-manifold and discrete Voronoi approximation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1938–1944, Sep. 2015.
[5] H. I. Choi, S. W. Choi, H. P. Moon, and N.-S. Wee, "New algorithm for medial axis transform of plane domain," *Graph. Models Image Process.*, vol. 59, no. 6, pp. 463–483, 1997.
[6] D. T. Lee, "Medial axis transformation of a planar shape," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 4, no. 4, pp. 363–369, Jul. 1982.
[7] K. Mehlhorn and S. Schirra, "Geometric computing with CGAL and LEDA," in *Curves and Surface Design: St. Malo 1999*, Nashville, TN, USA: Vanderbilt Univ. Press, 2000, pp. 277–286.
[8] C. Yap and T. Dube, "The exact computation paradigm," in *Computing in Euclidean Geometry*, D.-Z. Du and F.-K. Hwang, Eds. Singapore: World Scientific, 1995, pp. 452–492.
[9] A. Fabri, G.-J. Giezeman, L. Kettner, S. Schirra, and S. Schönherr, "On the design of CGAL, a computational geometry algorithms library," *Softw. Pract. Exp.*, vol. 30, no. 11, pp. 1167–1202, 2000.
[10] D. Salesin, J. Stolfi, and L. Guibas, "Epsilon geometry: Building robust algorithms from imprecise computations," in *Proc. 5th Annu. Symp. Comput. Geom.*, 1989, pp. 208–217.
[11] H. Brönnimann, C. Burnikel, and S. Pion, "Interval arithmetic yields efficient dynamic filters for computational geometry," *Discr. Appl. Math.*, vol. 109, nos. 1–2, pp. 25–47, 2001.
[12] K. Sugihara and M. Iri, "Construction of the Voronoi diagram for 'one million' generators in single-precision arithmetic," *Proc. IEEE*, vol. 80, no. 9, pp. 1471–1484, 1992.
[13] K. Sugihara and M. Iri, "A robust topology-oriented incremental algorithm for Voronoi diagrams," *Int. J. Comput. Geom. Appl.*, vol. 4, no. 2, pp. 179–228, 1994.
[14] K. Sugihara, M. Iri, H. Inagaki, and T. Imai, "Topology-oriented implementation - an approach to robust geometric algorithms," *Algorithmica*, vol. 27, no. 1, pp. 5–20, 2000.
[15] L. J. Latecki and R. Lakämper, "Convexity rule for shape decomposition based on discrete contour evolution," *Comput. Vis. Image Understanding*, vol. 73, no. 3, pp. 441–454, 1999.
[16] Z. Ren, J. Yuan, and W. Liu, "Minimum near-convex shape decomposition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 10, pp. 2546–2552, Oct. 2013.
[17] J. Serra, *Image Analysis and Mathematical Morphology*. San Diego, CA, USA: Academic Press, 1982.
[18] H. I. Choi, S. W. Choi, and H. P. Moon, "Mathematical theory of medial axis transform," *Pacific J. Math.*, vol. 181, no. 1, pp. 57–88, 1997.
[19] C. K. Yap, "An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments," *Discr. Comput. Geom.*, vol. 2, no. 1, pp. 365–393, 1987.
[20] R. T. Farouki and J. K. Johnstone, "The bisector of a point and a plane parametric curve," *Comput. Aided Geom. Des.*, vol. 11, no. 2, pp. 117–151, 1994.
[21] R. T. Farouki and R. Ramamurthy, "Specified-precision computation of curve/curve bisectors," *Int. J. Comput. Geom. Appl.*, vol. 8, nos. 5/6, pp. 599–617, 1998.
[22] J.-K. Seong, E. Cohen, and G. Elber, "Voronoi diagram computations for planar NURBS curves," in *Proc. Symp. Solid Phys. Modeling*, 2008, pp. 67–77.
[23] K. Tang and Y.-J. Liu, "Dynamic medial axes of planar shapes," in *Proc. Comput. Graph. Int.*, 2006, pp. 460–468.
[24] M. I. Karavelas, "A robust and efficient implementation for the segment Voronoi diagram," in *Proc. Int. Symp. Voronoi Diagrams Sci. Eng.*, 2004, pp. 51–62.
[25] S. Hertel and K. Mehlhorn, "Fast triangulation of simple polygons," in *Proc. Int. FCT-Conf. Fundam. Comput. Theory*, 1983, pp. 207–218.
[26] A. M. Bruckstein, G. Shapiro, and D. Shaked, "Evolutions of planar polygons," *Int. J. Pattern Recog. Artif. Intell.*, vol. 9, no. 6, pp. 991–1014, 1995.
[27] J. R. Shewchuk, "Adaptive precision floating-point arithmetic and fast robust geometric predicates," *Discr. Comput. Geom.*, vol. 18, no. 3, pp. 305–363, 1997.
[28] M. Held, "VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments," *Comput. Geom.: Theory Appl.*, vol. 18, no. 2, pp. 95–123, 2001.
[29] M. Held and S. Huber, "Topology-oriented incremental computation of Voronoi diagrams of circular arcs and straight-line segments," *Comput.-Aided Des.*, vol. 41, no. 5, pp. 327–338, 2009.

Yong-Jin Liu received the BEng degree from the Tianjin University, China, in 1998, and the PhD degree from the Hong Kong University of Science and Technology, Hong Kong, China, in 2004. He is an associate professor with the TNList, Department of Computer Science and Technology, Tsinghua University. His research interests include computational geometry, computer graphics, pattern analysis, and computer-aided design. He is a member of the IEEE.

Cheng-Chi Yu received the BEng degree from the Beijing University of Posts and Telecommunications, China, in 2013. He is a master student with the TNList, Department of Computer Science and Technology, Tsinghua University. His research interests include computatonal geometry, computer graphics, and image processing.

Min-Jing Yu received the BEng degree from the Wuhan University, China, in 2014, and is currently working toward the PhD degree with the TNList, Department of Computer Science and Technology, Tsinghua University. Her research interests include computer graphics, cognitive computation, and computer vision.

Kai Tang received the BEng degree from the Nanjing Institute of Technology, China, in 1982, the MSc degree in information and control engineering, in 1986, from the University of Michigan, and the PhD degree in computer engineering also from the University of Michigan, in 1990. He is a professor with the Department of Mechanical and Aerospace Engineering, Hong Kong University of Science and Technology. His research interests concentrate on designing efficient and practical algorithms for solving real-world computational and geometric problems.

Deok-Soo Kim received the BEng from the Hanyang University, Korea, in 1982, the MSc from the New Jersey Institute of Technology, in 1985, and the PhD degree from the University of Michigan, in 1990. He is a professor with the Voronoi Diagram Research Center and Department of Mechanical Engineering, Hanyang University, Seoul, Korea. He studies the Voronoi diagram of various kinds for both practical and theoretical view points, including discovering applications of Voronoi diagrams in engineering and science.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.