Constructing intrinsic delaunay triangulations from the dual of geodesic voronoi diagrams

Liu, Yong-Jin; Fan, Dian; Xu, Chun-Xu; He, Ying

2017

Liu, Y.-J., Fan, D., Xu, C.-X., & He, Y. (2017). Constructing intrinsic delaunay triangulations from the dual of geodesic voronoi diagrams. *ACM Transactions on Graphics*, 36(2), 15-.

<https://hdl.handle.net/10356/87052>

<https://doi.org/10.1145/2999532>

© 2017 Association for Computing Machinery. This is the author created version of a work that has been peer reviewed and accepted for publication by *ACM Transactions on Graphics*, Association for Computing Machinery. It incorporates referee's comments but changes resulting from the publishing process, such as copyediting, structural formatting, may not be reflected in this document. The published version is available at: [<http://dx.doi.org/10.1145/2999532>].

Downloaded on 26 Apr 2023 09:14:29 SGT

Constructing Intrinsic Delaunay Triangulations from the Dual of Geodesic Voronoi Diagrams

YONG-JIN LIU and DIAN FAN and CHUN-XU XU

Tsinghua University

and

YING HE

Nanyang Technological University

Intrinsic Delaunay triangulation (IDT) naturally generalizes Delaunay triangulation from $\mathbb{R}^2$ to curved surfaces. Due to many favorable properties, the IDT whose vertex set includes all mesh vertices is of particular interest in polygonal mesh processing. To date the only way for constructing such IDT is the edge-flipping algorithm, which iteratively flips non-Delaunay edges to become locally Delaunay. Although this algorithm is conceptually simple and it guarantees to terminate in finite steps, it has no known time complexity and may also produce triangulations containing faces with only two edges. This paper develops a new method to obtain proper IDTs on manifold triangle meshes. We first compute a geodesic Voronoi diagram (GVD) by taking all mesh vertices as generators and then find its dual graph. The sufficient condition for the dual graph to be a proper triangulation is that all Voronoi cells satisfy the so-called closed ball property. To guarantee the closed ball property everywhere, a certain sampling criterion is required. For Voronoi cells that violate the closed ball property, we fix them by computing topologically safe regions, in which auxiliary sites can be added without changing the topology of the Voronoi diagram beyond them. Given a mesh with n vertices, we prove that by adding at most $O(n)$ auxiliary sites, the computed GVD satisfies the closed ball property, hence its dual graph is a proper IDT. Our method has a theoretical worst-case time complexity $O(n^2 + tn \log n)$, where t is the number of obtuse angles in the mesh. Computational results show that it empirically runs in linear time on real-world models.

Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

General Terms: Algorithms

This work was supported by the National Key Research and Development Plan (2016YFB1001202), Natural Science Foundation of China (61432003, 61521002, 61661130156) and Royal Society-Newton Advanced Fellowship.

Authors' addresses: Y.-J. Liu, D. Fan and C.-X. Xu, TNList, Department of Computer Science and Technology, Tsinghua University, Beijing, China; email: liuyongjin@tsinghua.edu.cn; Y. He, School of Computer Engineering, Nanyang Technological University, Singapore; email: yhe@ntu.edu.sg.

Additional Key Words and Phrases: Intrinsic Delaunay triangulation, geodesic Voronoi diagram, duality, the closed ball property

ACM Reference Format:

Liu, Y.-J., Fan, D., Xu, C.-X., and He, Y. 2016. Constructing intrinsic Delaunay triangulations from the dual of geodesic Voronoi diagrams. *ACM Trans. Graph.* 28, 4, Article 106 (August 2016), 15 pages.

1. INTRODUCTION

A Delaunay triangulation for a set $\mathbf{P}$ of points in $\mathbb{R}^2$ is a triangulation such that no point of $\mathbf{P}$ is inside the circumcircle of any triangle in the triangulation. It is well known that Delaunay triangulations tend to avoid skinny triangles, since they maximize the minimum angle of all the angles of the triangles in the triangulation. Although Delaunay triangulations in Euclidean spaces are well understood [Okabe et al. 2000], *intrinsic* Delaunay triangulations (IDT) on manifold domains have received less attention. Using the closed ball property [Edelsbrunner and Shah 1997], Dyer et al. [2008] proposed adaptive sampling criteria for constructing intrinsic Voronoi diagram and its dual Delaunay triangulation on smooth 2-manifolds. Recently, Boissonnat et al. [2013] proposed an algorithm for constructing IDT on smooth closed submanifolds of Euclidean spaces. Both methods are based on *convex neighborhood*, which, in general, is an extremely small region around a point on smooth manifolds. As a result, in spite of their important theoretical values, they are not practical for piecewise linear surfaces, which are dominant in digital geometry processing.

In this paper, we focus on a special type of IDT defined on manifold triangle meshes, where the IDT's vertex set includes all mesh vertices. Such an IDT was first studied by Rivin [1994], who defined IDT edges using geodesic paths and replaced circumcircles by geodesic circumcircles. As pointed out by Bobenko and Springborn [2007], this kind of IDT is highly desired to digital geometry processing, since the classic cotangent Laplace-Beltrami operator (LBO) has non-negative weights w_{ij} , if and only if the underlying triangulation is Delaunay.

To date, the only practical algorithm for computing the above-mentioned IDTs is the edge-flipping algorithm [Bobenko and Springborn 2007; Fisher et al. 2007; Indermitte et al. 2001], which iteratively flips the non-Delaunay edge to become locally Delaunay. Bobenko and Springborn [2007] proved that the edge flipping algorithm terminates in a finite number of steps, therefore the intrinsic Delaunay tessellation exists. They also proved the uniqueness of the intrinsic Delaunay tessellation. The edge-flipping algorithm is conceptually simple and easy to implement. However, it

has no known time complexity and may also produce self-loops, leading to faces with only two edges.

This paper presents a new method with bounded time complexity and it can guarantee the computed IDTs are free of self-loops. Our idea is to construct IDT via the dual graph of geodesic Voronoi diagrams (GVD). Dyer et al. [2007b] established the duality between GVD and IDT on 2-manifold meshes and showed that if a GVD satisfies the closed ball property [Edelsbrunner and Shah 1997], the dual graph is a *proper*¹ IDT.

Our algorithm is the first-of-its-kind to compute GVDs on meshes which are guaranteed to satisfy the closed ball property. Given a mesh with n vertices, our method first computes the GVD by taking all vertices as sites. For each Voronoi cell that violates the closed ball property, it computes a topologically safe region, in which auxiliary sites can be added without changing the topology beyond that cell. We prove that by adding at most $O(n)$ auxiliary sites, the computed GVD satisfies the closed ball property, hereby it has a proper dual IDT. Moreover, thanks to the bounded time complexity of computing GVD [Liu et al. 2011; Liu and Tang 2013], our method has a theoretical worst-case time complexity $O(n^2 + tn \log n)$, where t is the number of obtuse angles in the mesh. We observe that many real-world models are far from their Delaunay triangulations, thus, it takes the edge-flipping algorithm many iterations to converge. In contrast, the performance of our method is not sensitive to the number of non-Delaunay edges and it empirically runs in linear time $O(n)$ on these models.

In addition to our theoretical contributions, the proper IDTs produced by our method are favorable to practical applications. Take the conformal parameterization [Fisher et al. 2007] as an example. The IDTs produced by the edge-flipping algorithm may contain degree-1 and/or 2 vertices. As a result, the injectivity of a discrete conformal mapping is lost, leading to rendering artifacts in texture mapping and also large local distortion. In contrast, our IDTs are proper and can be represented by any simplicial complex based data structure, making them an ideal input to the existing digital geometry processing pipeline.

The remaining of the paper is organized as follows: Section 2 reviews the mathematical background and highlights the fundamental differences between Delaunay triangulations in $\mathbb{R}^2$ and intrinsic Delaunay triangulations. Section 3 introduces the key ideas of our method. Section 4 presents our algorithm in details followed by the correctness proof and complexity analysis in Section 5. Section 6 reports the experimental results and compares IDTs with other Delaunay structures. Finally, Section 7 concludes the paper. To ease reading, we list the main notations in Table I and present the long proofs in Supplementary Material.

2. PRELIMINARIES

2.1 Geodesic Paths, Geodesic Triangles and Geodesic Circles

Let M be a manifold triangle mesh, and V, E, F the set of vertices, edges and faces of M , respectively. Every interior point of M has a neighborhood which is isometric to the neighborhood of either a point on an Euclidean plane or the apex of a Euclidean cone.

Consider two points $p, q \in M$. A geodesic path between p and q , denoted by $\gamma(p, q)$, is the *locally* shortest path between them. Mitchell et al. [1987] showed that the general form of a geodesic path γ is an alternating sequence of vertices (possibly empty) and

Table I. Main notations.

M	a manifold triangle mesh
V, E, F	the sets of vertices, edges and faces of M
$n(= V)$	the number of vertices
$v, v_i, v_j, \dots$	mesh vertices
$e, e_{ij} = \{v_i, v_j\}$	mesh edges
$f, f_{ijk} = \{v_i, v_j, v_k\}$	triangular faces
$p, q, p_i, q_i, \dots$	points on M
$\gamma(p, q)$	the geodesic path between p and q
$d(p, q)$	the geodesic distance between p and q
$D(p, r)$	the geodesic disk of radius r centered at p
$GVD(P)$	the geodesic Voronoi diagram of a set P of sites
$\Gamma(B_P, C_P)$	the topological representation of $GVD(P)$
B_P	the set of symbolic Voronoi edges
C_P	the set of symbolic Voronoi cells
$\beta(p, q)$	the bisector of sites p and q
$b(p, q)$	the symbolic representation of a Voronoi edge
$\tilde{b}(p, q)$	the geometric realization of $b(p, q)$
$C(p_i)$	the symbolic representation of a Voronoi cell
$\tilde{C}(p_i)$	the geometric realization of $C(p_i)$
$pb(p)$	the symbolic representation of a pseudo bisector
$\tilde{pb}(p)$	the geometric realization of $pb(p)$
$\delta_{\max}(\tilde{C}(p))$	the maximal value of δ so that the δ -offset $\Theta(\tilde{C}(p), \delta)$ does not contain any site in P and any Voronoi vertex that is not in $\tilde{C}(p)$.
$\delta_{\max}(GVD(P))$	the minimal $\delta_{\max}$ for all Voronoi cells
$\varepsilon(\delta)$	the radius of the ε -neighborhood $D(p, \varepsilon)$ for any $0 < \delta \leq \delta_{\max}(GVD(P))$
$\hat{\delta}_{\max}(GVD(P))$	the value for a strong δ -offset defined in Eq. (S6) in Supplementary Material
single-order neighbor	two sites whose Voronoi cells share exactly one Voronoi edge
multiple-order neighbor	two sites whose Voronoi cells share at least two Voronoi edges
$IDT(M)$	the intrinsic Delaunay triangulation of M
Ξ	the set of g -edges
Γ	the set of geodesic triangles
$\xi \in \Xi$	g -edges
$\tau, \tau_i, \tau_j \in \Gamma$	geodesic triangles

edge sequences such that the unfolded image of the path γ along any edge sequence is a straight line segment and the angle of the path at a vertex is greater than or equal to π . The vertices with cone angles more than 2π are called *saddle vertices*, which play a critical role in geodesic computation [Ying et al. 2013; Xu et al. 2015]. We denote by $d(p, q)$ the geodesic distance between p and q , and $\beta(p, q)$ the bisector of p and q .

In a smooth, simply connected surface S with negative Gaussian curvature everywhere, the geodesic path $\gamma(p, q)$ is unique for any pair of points $p, q \in S$. However, in general, geodesics are not unique for regions with positive Gaussian curvature and/or non-trivial topology. See Figure 1(a)-(c). It is worth noting that discrete geodesic paths have similar properties, since Gauss-Bonnet theorem also holds on piecewise linear surfaces. A geodesic γ joining two points p and q is *minimal* if its length is smaller than or equal to the length of any curve joining p and q .

DEFINITION 1 GEODESIC TRIANGLE. A *geodesic triangle* $\tau \subset M$ is a simply connected domain whose boundary $\partial\tau$ has three geodesic paths. Each geodesic path is called a *g-edge* and the endpoints of a *g-edge* are called *g-vertices*.

¹A triangulation is *proper* if it is a realization of a simplicial complex. See [Dyer et al. 2007b]. All Delaunay triangulations in $\mathbb{R}^2$ are proper.

Fig. 1. Geodesic paths & geodesic triangles. (a) Domain D_1 is simply connected and has non-positive Gaussian curvature everywhere. There is a *unique* geodesic path between any pair of distinct points in D_1 . (b) Although domain D_2 is also simply connected, it has a part with positive Gaussian curvatures. As a result, the geodesic paths in D_2 are not unique. (c) Domain D_3 has negative Gaussian curvature everywhere. However, as its fundamental group is non-trivial, the geodesic paths in D_3 are not unique either. (d) The red region $\tau = (p_1, p_2, p_3)$ is a geodesic triangle on the torus M . However, the complement $M \setminus \tau$, although having three geodesic sides, is not a geodesic triangle, since it is not simply connected. (e) Three geodesic paths $\gamma(p_4, p_5)$, $\gamma(p_5, p_6)$ and $\gamma(p_6, p_4)$ wrap the cylinder. Although each colored region has three geodesic sides, they are not geodesic triangles, since it does not bound a simply connected region.

Fig. 2. Geodesic circumcircles. (a) Geodesic triangle $\triangle p_1 p_2 p_3$ has no geodesic circumcircle, since the bisector $\beta(p_1, p_2)$ (shown in blue) does not meet the other bisector $\beta(p_2, p_3)$. (b) Since $\beta(p_4, p_5)$ (blue) and $\beta(p_5, p_6)$ (red) intersect twice, there are two geodesic circles passing through three points p_4 , p_5 and p_6 ; only one of them is the geodesic circumcircle of the geodesic triangle $\triangle p_4 p_5 p_6$.

On $\mathbb{R}^2$, any three intersecting lines form a triangle. However, not all three intersecting geodesics on a mesh form a geodesic triangle. See Figure 1(d)-(e).

DEFINITION 2 GEODESIC DISK & GEODESIC CIRCLE. A *geodesic disk* of radius r centered at a point $p \in M$, denoted by $D(p, r)$, consists of all points whose geodesic distances to p does not exceed r , i.e., $D(p, r) = \{q \in M : d(p, q) \leq r\}$. If a geodesic disk $D(p, r)$ is simply connected, its boundary $\partial D(p, r)$ is called a *geodesic circle*. A geodesic circle ∂D that circumscribes a geodesic triangle τ , $\tau \subset D$, is called a *geodesic circumcircle*.

In $\mathbb{R}^2$, three distinct points not lying on a line define a non-degenerate triangle that has a unique circumcircle. However, on a curved surface, not every geodesic triangle has a geodesic circumcircle. For the case that the geodesic circumcircle exists, there may be multiple geodesic circles passing through the three vertices of the geodesic triangle. See Figure 2.

2.2 Intrinsic Delaunay Triangulations on Meshes

The definition of intrinsic Delaunay triangulation on 2-manifold meshes is due to Rivin [1994], who generalized the planar Delaunay condition (i.e., a circle circumscribing any Delaunay triangle does not contain any input points in its interior), by requiring the empty *geodesic circumcircle* property.

DEFINITION 3 INTRINSIC DELAUNAY TRIANGULATION. The *intrinsic Delaunay triangulation (IDT)* on M , denoted by $IDT(M) = (V, \Xi, \Gamma)$, is a tessellation of M such that

- the vertex set of $IDT(M)$ equals V ;
- every edge ξ in Ξ is a geodesic path on M , i.e., a *g-edge*;
- each face $\tau \in \Gamma$ is a geodesic triangle, which has a geodesic circumcircle containing no mesh vertices in its interior.

Then Dyer et al. [2007b] defined *proper IDT* as follows:

DEFINITION 4 PROPER IDT. An IDT on M is *proper* if it is the realization of a simplicial complex.

Bobenko and Springborn [2007] showed the edge-flipping algorithm terminates in finite steps, implying the existence of intrinsic Delaunay triangulations. They also proved the uniqueness of Delaunay tessellation (whose faces are general but not always triangular). The Delaunay triangulation can be obtained by triangulating the non-triangular faces.

2.3 Geodesic Voronoi Diagrams

DEFINITION 5 GEODESIC VORONOI DIAGRAM. Let $P = \{p_1, p_2, \dots, p_m\}$ be a set of points on M . For a site p_i , the *Voronoi cell* $\tilde{C}(p_i)$ consists of all points whose geodesic distances to p_i are less than or equal to their distances to any other site, i.e., $\tilde{C}(p_i) = \{q \in M : d(p_i, q) \leq d(p_j, q), \forall i \neq j\}$. The *geodesic Voronoi diagram (GVD)* of P is the set $\{\tilde{C}(p_1), \tilde{C}(p_2), \dots, \tilde{C}(p_m)\}$.

In Sections 3-5, we assume that the points of P are in *general* positions, i.e., no four or more points are on the same geodesic circumcircle, hereby the intersection of four or more Voronoi cells is

Fig. 3. Bisectors and Voronoi cells on a 2-manifold mesh are significantly different from their Euclidean counterparts. Row 1: There is a saddle vertex (yellow) on the geodesic path $\gamma(p, q)$ (cyan). As a result, the bisector $\beta(p, q)$ (blue) consists of both line segments and hyperbolic segments. Row 2: $C(p_3)$ (green) is of genus-1, and $C(p_1)$ (red) and $C(p_2)$ (yellow) are of genus-0 but with multiple boundaries. Row 3: A GVD with 8 sites p_i , $i = 1, \dots, 8$ on the Two-hole Torus model. Note that each of $C(p_5)$, $C(p_6)$, $C(p_7)$ and $C(p_8)$ has multiple boundaries. $C(p_1)$ and $C(p_2)$ share two common Voronoi edges, and so do the other pairs of Voronoi cells $(C(p_2), C(p_3))$, $(C(p_3), C(p_4))$, $(C(p_1), C(p_4))$ and $(C(p_2), C(p_4))$.

an empty set. In Supplementary Material, we extend the proof of main results in this paper to include degenerate cases. In Section 6.3, the implementation issue of handling degenerate cases is discussed.

It is well known that Voronoi cells in $\mathbb{R}^2$ are convex and simply connected. However, this property does not hold for geodesic Voronoi diagrams [Liu et al. 2011; Xu et al. 2014; Liu 2015]. Although a geodesic Voronoi cell is still connected, it may contain multiple boundaries and/or handles. See Figure 3 (rows 2 and 3). Moreover, bisectors on triangle meshes, in general, consist of line segments and hyperbolic segments [Liu et al. 2011]. See Figure 3 (row 1). For a special case that P includes all mesh vertices, all Voronoi edges of $GVD(P)$ are line segments [Dyer et al. 2007b; Liu et al. 2011].

2.4 The Closed Ball Property & Dual Proper IDT

In Euclidean domains, the dual graph of the Voronoi diagram of a point set P is a *proper* Delaunay triangulation; therefore, one can adopt the algorithms for constructing Voronoi diagram to obtain Delaunay triangulation, and vice versa. However, in general manifold domains, the dual *proper* IDT exists only if the GVD satisfies the closed ball property [Edelsbrunner and Shah 1997].

The closed ball property consists of three conditions:

- (1) **Disk condition:** each Voronoi cell is homeomorphic to a planar disk;
- (2) **2-cell intersection condition:** the intersection of any two Voronoi cells is either empty or a single Voronoi edge;
- (3) **3-cell intersection condition:** the intersection of any three Voronoi cells is either empty or a single Voronoi vertex.

Throughout this paper, we assume the point set P includes all mesh vertices in M and the points of P satisfy the general position condition. Dyer et al. [2007b] showed that if there are at least four distinct sites in $GVD(P)$ and both the disk condition and 2-cell intersection conditions are satisfied, then the 3-cell intersection condition is redundant.

THEOREM 6. [Dyer et al. 2007b] *The IDT is proper if and only if the dual GVD satisfies the closed ball property.*

By Theorem 6, the key to obtain a proper IDT is to construct a GVD satisfying the closed ball property. However, Dyer et al. [2007b] did not provide any algorithm. Our paper fills the gap by developing a practical algorithm to compute such GVDs.

Note that Edelsbrunner and Shah [1997] introduced the closed ball property for triangulating abstract topological spaces. Recently, Dyer et al. [2015] studied a natural intrinsic definition of geometric simplices in Riemannian manifolds of arbitrary finite dimension, and exploited these simplices to obtain criteria for triangulating compact Riemannian manifolds.

3. KEY IDEAS

3.1 Motivation

Flipping edges and computing the dual graph of Voronoi diagrams are two commonly used techniques for constructing Delaunay triangulations in $\mathbb{R}^2$. For IDTs, the edge flipping algorithm [Indermitte et al. 2001; Fisher et al. 2007] is the only known algorithm so far. It is conceptually simple and easy to implement, however, it does not have a known time complexity and the computed IDT may not be proper, due to faces with only two edges. Motivated by Theorem 6, we take the other direction to construct IDT, i.e., computing the dual of geodesic Voronoi diagrams satisfying the closed ball property.

We consider only the disk condition and the 2-cell intersection condition in our algorithm, since all the models we are dealing with have more than four vertices. Observe that the closed ball property often fails in regions with low sampling density. Thus, our idea is to add auxiliary sites to those Voronoi cells that violate the disk condition and 2-cell intersection condition. However, one has to be careful when adding an auxiliary site to a Voronoi diagram. Let $V(q)$ be a Voronoi cell violating the closed ball property. Adding a misplaced auxiliary site s to $V(q)$, though fixing the *original* Voronoi cell $V(q)$, may cause problems to the new Voronoi cell $V(s)$. See two such examples in Figures 4 and 5.

To avoid the endless loop for adding auxiliary sites and also ease analysis, we introduce a novel concept, called *topologically safe site*, where adding such a site to a GVD does not change the topology beyond the Voronoi cell containing the site. In the following, we introduce the topological representation of GVD and then prove the existence of topologically safe sites.

3.2 Topological Representation

Before defining topologically safe site, we introduce the *topological* representation of Voronoi cells. Note that a Voronoi cell in the GCD can be topologically non-trivial due to multiple boundaries.

Fig. 4. Illustration of our strategy to ensure the disk condition. *Row 1:* Voronoi cell $C(p_1)$ violates the closed ball property, due to two boundaries and one pseudo bisector (green, to be defined in Section 4.3). *Row 2:* Although adding an auxiliary site q on the pseudo bisector makes $C(p_1)$ a topological disk, the new cell $C(q)$ is not simply connected and it also contains one pseudo bisector. *Row 3:* Adding two topologically safe sites q_1 and q_2 (see Section 4.3 for details), our method fixes $C(p_1)$ and guarantees that the disk condition holds for $C(q_1)$ and $C(q_2)$. Moreover, the topologies of $C(p_2)$, $C(p_3)$ and $C(p_4)$ remain unchanged.

Fig. 5. Illustration of our strategy to ensure the 2-cell intersection condition. (a) All cells are topologically equivalent to a disk. However, $C(p_1)$ and $C(p_4)$ share two common Voronoi edges (pink), violating the 2-cell intersection condition. (b) Although adding an auxiliary site q (cyan) at one of the shared edges can separate $C(p_1)$ and $C(p_4)$, the new Voronoi cell $C(q)$ is not simply connected and it contains a pseudo-bisector (green), violating the disk condition. (c) For each common Voronoi edge, our method adds two auxiliary sites in the topologically safe region (see Section 4.4 for details). As a result, it not only separates the problematic cells $C(p_1)$ and $C(p_4)$, but also guarantees that all new cells $C(q_i)$, $1 \leq i \leq 4$, are topological disks and they satisfy the 2-cell intersection condition.

As Figure 8(b) shows, there are three boundaries in the Voronoi cell of the “crotch” vertex in the Pant model. For a site s , we represent the Voronoi cell $C(s)$ using its disjoint boundaries, i.e., $C(s) = \{l_1, l_2, \dots, l_h\}$. Each boundary is a loop $l_i = (b_{i1}, b_{i2}, \dots, b_{i_{l_i}})$, which is an *ordered* list of Voronoi edges so that the interior of $C(s)$ is always on the left side when walking from b_{ij} to $b_{i,j+1}$ ($b_{i,l_i+1} = b_{i1}$). See Figure 6 for an illustration. Since each Voronoi edge is a trimmed bisector of two sites, we represent the Voronoi edge $b(p, q)$ by an *ordered* pair² of the indices of sites p and q . A Voronoi cell is called *simple* if it has only one boundary, i.e., its cardinality $|C(s)| = 1$; otherwise it is called *non-simple*.

Given a set of sites $P = \{p_i | p_i \in M\}_{i=1}^m$, we represent the geodesic Voronoi diagram of P by a 2-tuple, $GVD(P) = (P, \Gamma(B_P, C_P))$, where the topological representation $\Gamma(B_P, C_P)$ consists of the sets of symbolic Voronoi edges B_P and Voronoi cells C_P . With the topological representation of a GVD, its geometry is readily available from the positions of sites P , using the algorithm `genus_r_Voronoi_diagram(M, P)` in [Liu et al. 2011].

²Since there may be more than one shared Voronoi edge between two sites, we allow multiple occurrences of an ordered pair.

Fig. 6. The topological representation of a Voronoi cell $C(s)$ with three disjoint boundaries, $C(s) = \{l_1, l_2, l_3\}$. Each boundary is an ordered list of Voronoi edges, $l_1 = (b_{11}, b_{12}, b_{13}, b_{14})$, $l_2 = (b_{21}, b_{22}, b_{23})$ and $l_3 = (b_{31}, b_{32}, b_{33})$.

Throughout the paper, we use $\sim$ to distinguish an entity in its *topological* representation and *geometric* realization. For example, the topological representation $b(p, q)$ is an ordered pair of the indices of p and q , whereas the geometric representation $\tilde{b}(p, q)$ is a polyline on the mesh. Similarly, $\tilde{l}_i$ and $\tilde{C}(s)$ are geometric realizations of boundary l_i and Voronoi cell $C(s)$, respectively.

3.3 Topologically Safe Sites

DEFINITION 7 TOPOLOGICALLY SAFE SITE. Let P be a set of sites on M including all mesh vertices, and $GVD(P)$ the corresponding geodesic Voronoi diagram. For a subset $Q \subseteq P$, let $B_Q(P) \triangleq \{b(r, s) : b(r, s) \in B_P \text{ and } r, s \in Q\}$ be the set of Voronoi edges which are elements of B_P and are generated by sites in Q . Consider a new site $q \notin P$, which is inside a Voronoi cell, say $q \in C(p)$. We say q is *topologically safe* to $GVD(P)$, if $B_{P \setminus \{p\}}(P \cup \{q\}) = B_{P \setminus \{p\}}(P)$.

Intuitively speaking, the new site q is topologically safe to $GVD(P)$ if adding q to the Voronoi diagram does not destroy any existing Voronoi edges (in terms of topological representation) other than the edges of the Voronoi cell $C(p)$ that contains q , i.e., $q \in C(p)$. See Figure 7 for examples of topologically safe and unsafe sites.

Fig. 7. Topologically safe and unsafe sites. Given four sites $P = \{p_1, p_2, p_3, p_4\}$ in two different positions, the corresponding GVDs have the same topological representation, i.e., $B_P = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 4)\}$. Row 1: The site p_5 is topologically safe to $GVD(P)$, since $B_{P \setminus \{p_4\}}(P \cup \{p_5\}) = B_{P \setminus \{p_4\}}(P) = \{(1, 2), (1, 3), (2, 3)\}$; that is, adding p_5 does not destroy any existing Voronoi edge $b(p_i, p_j)$ for $i \neq 4, j \neq 4$. Row 2: The site p_6 is not topologically safe to $GVD(P)$, since $B_{P \setminus \{p_4\}}(P \cup \{p_6\}) = \{(1, 2), (1, 3)\}$ and $B_{P \setminus \{p_4\}}(P) = \{(1, 2), (1, 3), (2, 3)\}$. Visually, adding p_6 to the Voronoi diagram destroys an existing Voronoi edge $b(p_2, p_3)$, which does not belong to $C(p_4)$. The new Voronoi edges are colored in red and the existing Voronoi edges in black.

To quantitatively characterize the topologically safe region and find topologically safe sites, we define the δ -offset for a Voronoi cell.

DEFINITION 8 δ -OFFSET. Consider a geometric realization of Voronoi cell $\tilde{C}(p)$ with boundaries $\tilde{C}(p) = (\tilde{l}_1, \tilde{l}_2, \dots, \tilde{l}_h)$. Let $\Theta(\tilde{C}(p), \delta) = \{x : d(x, y) < \delta, y \in \tilde{b}_i, \tilde{b}_i \subseteq \tilde{l}_1 \cup \tilde{l}_2 \cup \dots \cup \tilde{l}_h\}$ be the offset of the boundary of $\tilde{C}(p)$. $\delta_{\max}(\tilde{C}(p))$ is the maximal value of δ so that $\Theta(\tilde{C}(p), \delta)$ does not contain any site in P and any Voronoi vertex that is not in $\tilde{C}(p)$. For a $GVD(P)$, define

$$\delta_{\max}(GVD(P)) \triangleq \min_{p \in P} \delta_{\max}(\tilde{C}(p)), \quad (1)$$

the minimal $\delta_{\max}$ for all Voronoi cells.

The following proposition shows that each Voronoi cell has an ε -neighborhood, in which any point is topologically safe to the GVD. This allows us to add auxiliary sites to a Voronoi cell without changing the topology of its neighbors.

PROPOSITION 1 EXISTENCE OF ε -NEIGHBORHOOD. For any $0 < \delta \leq \delta_{\max}(GVD(P))$, there always exists an $\varepsilon > 0$, such that in each Voronoi cell $C(p)$, $D(p, \varepsilon) = \{x : x \in M \text{ and } d(p, x) \leq \varepsilon\}$ is simply connected and any point x in $D(p, \varepsilon)$ is topologically safe to $GVD(P)$. $D(p, \varepsilon)$ is called the ε -neighborhood of p .

The value of ε depends on δ . See Eq. (S1) in Supplementary Material. Figure 8(c) shows an example of the δ -offset and ε -neighborhood on the Pant model.

Fig. 8. Pseudo-bisectors occur in cylinder-shaped regions. There are one pseudo-bisector in $C(p)$ (see (a)) and two in $C(q)$ (see (b)). The bisectors and pseudo-bisectors are colored in blue and green respectively. (c) Outward δ -offset (green) of a Voronoi cell and its corresponding ε -neighborhood colored in orange.

Algorithm 1 Constructing Proper Intrinsic Delaunay Triangulation from the Dual of Geodesic Voronoi Diagram

Input: $M = (V, E, F)$, a 2-manifold triangle mesh

Output: $IDT(M)$, a proper IDT defined on M , whose vertex set contains all vertices in V

- 1: $GVD(V) = \text{compute_gvd}(M)$ (Procedure 2 in Section 4.2)
- 2: $GVD(P) = \text{ensure_disk_condition}(GVD(V))$, where $V \subseteq P$ (Procedure 3 in Section 4.3)
- 3: $GVD(P') = \text{ensure_2-cell_intersection_condition}(GVD(P))$, where $P \subseteq P'$ (Procedure 4 in Section 4.4)
- 4: $IDT(M) = \text{compute_dual_graph}(GVD(P'))$ (Procedure 5 in Section 4.5)

4. ALGORITHM

4.1 Overview

To facilitate the presentation, we assume the input 2-manifold mesh M is closed. In Supplementary Material, we extend our algorithm to handle meshes with boundaries. Our algorithm (ref. Algorithm 1) consists of four steps. First, taking all mesh vertices as sites, it computes the geodesic Voronoi diagram $GVD(V)$. Then, it checks the disk condition for all Voronoi cells. If a Voronoi cell, say $C(v_i)$, is not homeomorphic to a disk, the algorithm adds auxiliary sites in the ε -neighborhood of v_i to fix $C(v_i)$. Next, it checks the 2-cell intersection condition for all pairs of adjacent Voronoi cells. If two cells, say $C(v_i)$ and $C(v_j)$, share two or more Voronoi edges, the algorithm again adds auxiliary sites to reduce the number of common edges to one. Finally, since the updated GVD satisfies the closed ball property, it computes the dual graph, which is a proper IDT. Intuitively speaking, our method adaptively increases the sampling density for the regions where the closed ball property fails. Figure 9 illustrates the algorithmic pipeline using a toy cylinder model.

4.2 Computing GVD

Liu et al. [2011] presented a generic algorithm to construct GVD on manifold triangle meshes. For $m (\leq n)$ arbitrary sites placed on an n -vertex mesh M , the algorithm runs in $O(n^2 \log n)$ time. One, of course, can apply Liu et al.'s algorithm directly to our application. However, our scenario is slightly different in that we take *all*

Fig. 9. Algorithmic pipeline. (a) The cylinder mesh has 17 vertices (red). Taking all mesh vertices as sites, we compute the geodesic Voronoi diagram $GVD(V)$. However, $GVD(V)$ does not have a dual proper Delaunay triangulation, since Voronoi cell $C(v_1)$ is not simply connected and it also contains a pseudo-bisector (green). (b) After adding an auxiliary site q_1 (cyan) in the ε -neighborhood of site v_1 , both $C(v_1)$ and $C(q_1)$ become topological disks. However, they share two common Voronoi edges (purple). (c) To ensure the 2-cell intersection condition, we add another auxiliary site q_2 in the ε -neighborhood of v_1 . (d) Now the geodesic Voronoi diagram $GVD(P')$ satisfies the closed ball property, thus, its dual graph is a proper intrinsic Delaunay triangulation.

mesh vertices as sites. For better performance, we adapt Liu et al's algorithm as follows:

We assign all mesh vertices a distance value 0, since they are the source points. Then we apply the MMP algorithm to compute the geodesic distances for the points on mesh edges. After termination, the MMP algorithm partitions each mesh edge into disjoint intervals, called *windows*, where each window encodes the shortest paths coming from the same face sequence. Note that the original MMP algorithm must solve a quadratic equation to trim overlapped windows, due to the existence of hyperbolic bisectors. Fortunately, all mesh vertices are the sources in our case, hence, we solve only a linear equation for window trimming.

We are interested in the mesh edges which contain windows corresponding to *different* sources, since they are crossed by some Voronoi edges. Observe that a Voronoi vertex corresponds to three Voronoi edges. Therefore, we can locate Voronoi vertices by finding those triangles where three or more Voronoi edges meet. In lines 6-16 of Procedure 2, we compute the Voronoi edges with distinct endpoints. Then in lines 17-22, we compute the Voronoi edges which are self-loops.

For simplicity, the pseudo-code in Procedure 2 does not handle the case that a triangular face contains two or more Voronoi vertices. We refer readers to Algorithm 1³ in [Xu et al. 2014] for details.

4.3 Ensuring the Disk Condition

In $\mathbb{R}^2$, the term “bisector” refers to the set of points that are equidistant to two *distinct* sites. In contrast to the 2D counterpart, there are two types of bisectors on polyhedral surfaces: one is the conven-

tional bisector of two distinct sites, and the other corresponds to a *single* site, hereby called *pseudo-bisector*.

DEFINITION 9 PSEUDO-BISECTOR. *The pseudo-bisector of a site p consists of points $q \in \tilde{C}(p)$ such that there are two geodesic paths realizing the minimum distance between p and q .*

Pseudo-bisectors occur in cylinder-shaped regions. See Figure 8 (a) and (b). Although a Voronoi cell may have multiple pseudo-bisectors, the total number of pseudo-bisectors in $GVD(V)$ is bounded by the number of vertices in a mesh.

PROPOSITION 2 COMPLEXITY OF PSEUDO-BISECTORS. *The number of pseudo-bisectors in $GVD(V)$ is $O(n)$, where $n = |V|$ is the number of vertices in M .*

Observe that pseudo-bisectors are due to low sampling density. For a Voronoi cell $C(v_i)$, $v_i \in V$ with pseudo-bisectors, we add auxiliary sites in the ε -neighborhood of v_i to destroy all the pseudo bisectors in $C(v_i)$. Since those auxiliary sites are topologically safe to $GVD(V)$, the topologies of the Voronoi cells adjacent to $C(v_i)$ remain unchanged. As a result, we only need to ensure that the Voronoi cells of these auxiliary sites are topological disks.

We first compute the $\delta_{\max}$ -offset for $GVD(P)$ using Eq. (1), such that the corresponding $\varepsilon(\delta_{\max})$ -neighborhood contains topologically safe sites. Then we compute a *strong* δ -offset $\hat{\delta}(GVD(P)) \leq \delta_{\max}(GVD(P))$ using Eq. (S6) in Supplementary Material, such that pseudo-bisectors cannot appear in the Voronoi cells of newly added topologically safe sites in $\varepsilon(\hat{\delta})$ -neighborhood, which are sampled in the following way.

Figure 10 illustrates our strategy for eliminating pseudo-bisectors. The Voronoi cell $C(p)$ has multiple boundaries and it has two pseudo-bisectors. For each pseudo-bisector $\tilde{p}b_i(p)$, let $x_i \in \tilde{p}b_i$ be the point that minimizes the geodesic distance from p to $\tilde{p}b_i$. Since $x_i \in \tilde{p}b_i$, there are two minimal geodesics $\gamma(p, x_i)$

³Its implementation details are available at <http://cg.cs.tsinghua.edu.cn/people/~Yongjin/PGVD-pg2014-supplement.pdf>

Procedure 2 compute_gvd(M)

```

1: Taking all mesh vertices  $V$  as sources, apply the MMP algo-
   rithm to compute geodesic distances.
2: Put the triangles containing windows from exactly two differ-
   ent sources into a list  $BT$ .
3: Put the triangles containing windows from three or more dif-
   ferent sources into a list  $TT$ .
4: Create a Voronoi vertex list  $\Omega$  from  $TT$ , in which each Voronoi
   vertex  $\omega_i$  with sources  $(v_{i_1}, v_{i_2}, v_{i_3})$  corresponds to a triangle
    $f_i$  in  $TT$ .
5: Initialize a Voronoi edge list  $B_V = \emptyset$ .
6: // Compute Voronoi edges ended at distinct Voronoi vertices
7: for every  $w_i \in \Omega$  do
8:   Get the corresponding  $f_i$  from  $\Omega$ 
9:   for  $m = 1$  to 3 do
10:    if the Voronoi edge  $b(v_{i_m}, v_{i_{(m+1)\%3}}) \notin B_V$  then
11:      Starting from face  $f_i$ , compute the Voronoi edge
       $\tilde{b}(v_{i_m}, v_{i_{(m+1)\%3}})$  until reaching another face  $f_j$  con-
      taining a Voronoi vertex.
12:      Remove those triangles crossed by  $\tilde{b}(v_{i_m}, v_{i_{(m+1)\%3}})$ 
      from  $BT$ .
13:      Add  $b(v_{i_m}, v_{i_{(m+1)\%3}})$  into  $B_V$ .
14:    end if
15:  end for
16: end for
17: // Compute Voronoi edges which are self-loops
18: while  $BT$  is not empty do
19:   Take the first face  $f$  from  $BT$ . //  $f$  is crossed by only one
   Voronoi edge
20:   Starting from  $f$ , compute the Voronoi edge  $\tilde{b}'$  until getting
   back to  $f$ .
21:   Remove those triangles crossed by  $\tilde{b}'$  from  $BT$ .
22: end while
23: Extract the Voronoi cells  $C_V$  from the set of Voronoi edges
    $B_V$ .
24: return  $(B_V, C_V)$ , the topological representation of  $GVD(V)$ .

```

and $\gamma'(p, x_i)$ from p to x_i in $\tilde{C}(p)$. To eliminate a pseudo-bisector $\tilde{p}b_i(p)$, we add two auxiliary sites $a_i \in \gamma(p, x_i)$ and $b_i \in \gamma'(p, x_i)$ such that $d(p, a_i) = d(p, b_i) = \varepsilon(\hat{\delta})$.

Observe that for any point $y \in \tilde{p}b_i(p)$, the geodesic distance $d(y, p) > \max\{d(y, a_i), d(y, b_i)\}$. Therefore, adding a_i and b_i to the Voronoi diagram destroys the pseudo-bisector $\tilde{p}b_i(p)$. Moreover, both a_i and b_i are topologically safe sites, as they are within the ε -neighborhood of site p .

In fact, adding one auxiliary site a_i is sufficient to destroy the pseudo-bisector $\tilde{p}b_i$. We need the other auxiliary site b_i to ensure that there aren't new pseudo-bisectors in $\tilde{C}(a_i)$ and $\tilde{C}(b_i)$. In general, for a Voronoi cell $C(p)$ with k pseudo-bisectors, the following proposition shows that adding $2k$ auxiliary sites are sufficient to eliminate all the existing pseudo-bisectors. At the same time, no new pseudo-bisectors will be introduced.

PROPOSITION 3 ELIMINATING PSEUDO-BISECTORS. *Let $\tilde{C}(p)$ be a Voronoi cell with k pseudo-bisectors in $GVD(P)$, $V \subseteq P$. After adding $2k$ auxiliary sites $\{a_i, b_i\}_{i=1}^k$ to $\tilde{C}(p)$, the updated cells $\tilde{C}(p)$ and $\{\tilde{C}(a_i), \tilde{C}(b_i)\}_{i=1}^k$ are all topological disks. The topology of all other cells remains unchanged.*

Procedure 3 ensure_disk_condition($GVD(V)$)

```

1:  $P = V$ 
2: Compute  $\varepsilon(\hat{\delta}(GVD(V)))$  using Eqs. (S1, S6, S7) in Supple-
   mentary Material.
3: for every  $v \in V$  do
4:   if  $C(v)$  has two or more boundaries then
5:     Initialize sets of auxiliary sites  $A = \emptyset$  and  $B = \emptyset$ .
6:     for each pseudo-bisector  $\tilde{p}b_i \subset \tilde{C}(v)$  do
7:       Find  $x_i \in \tilde{p}b_i$  which minimizes the distance from  $v$  to
        $\tilde{p}b_i$ .
8:       Compute two minimal geodesics  $\gamma(v, x_i)$  and
        $\gamma'(v, x_i)$ .
9:       Find  $a_i \in \gamma$  and  $b_i \in \gamma'$  such that  $d(v, a_i) =$ 
        $d(v, b_i) = \varepsilon(\hat{\delta})$ .
10:       $A = A \cup \{a_i\}$  and  $B = B \cup \{b_i\}$ .
11:     end for
12:      $P = V \cup A$ .
13:     Locally update  $GVD(P)$ ,  $\hat{\delta}(GVD(P))$  and  $\varepsilon(\hat{\delta})$ .
14:     for each auxiliary site  $a_i \in A$  do
15:       if  $C(a_i)$  has two or more disjoint boundaries then
16:          $P = P \cup \{b_i\}$ .
17:         Locally update  $GVD(P)$ ,  $\hat{\delta}(GVD(P))$  and  $\varepsilon(\hat{\delta})$ .
18:       end if
19:     end for
20:   end if
21: end for
22: return  $GVD(P)$ , in which all Voronoi cells are topological
   disks.

```

Fig. 10. Voronoi cell $\tilde{C}(p)$ has two pseudo-bisectors. To destroy the pseudo-bisector $\tilde{p}b_i(p)$, we add two auxiliary sites a_i and b_i in the ε -neighborhood of p . See the main text for details.

Theoretically, $2k$ auxiliary sites are required to eliminate all pseudo-bisectors in $C(p)$. In practice, we adopt an incremental strategy in Procedure 3. We add the auxiliary site a_i first and then update the topological representations of $C(p)$ and $C(a_i)$. We add the other site b_i only when $C(a_i)$ has two or more boundaries.

Since there are $O(n)$ pseudo-bisectors in a mesh, Procedure 3 adds $O(n)$ auxiliary sites so that all Voronoi cells in $GVD(P)$, $V \subseteq P$, are topological disks.

4.4 Ensuring the 2-cell Intersection Condition

After eliminating pseudo-bisectors, all Voronoi cells in $GVD(P)$ are topological disks. Now we check the 2-cell intersection condition for all-pairs of adjacent Voronoi cells. Let B_P be the set of symbolic Voronoi edges in $GVD(P)$. Recall that in topological representation, the Voronoi edge $b(p_i, p_j)$ is an ordered pair $(\min\{I(p_i), I(p_j)\}, \max\{I(p_i), I(p_j)\})$, where $I(p)$ is the index of site p . We sort the Voronoi edges in B_P in ascending order by their first elements. If two ordered pairs have the same first element, the second element is used to break a tie. If a Voronoi edge

Fig. 11. Since each Voronoi cell $C(p_i)$, $i = 1, 2$, has two shared Voronoi edges b_{i1} and b_{i2} with $C(p)$, p_i is a multiple-order neighbor of p .

$b(p_i, p_j)$ appears more than once in B_P , the corresponding Voronoi cells $C(p_i)$ and $C(p_j)$ violate the 2-cell intersection condition. See Figure 11 for an example.

DEFINITION 10 NEIGHBORING SITE. p and q are neighboring sites if $C(p)$ and $C(q)$ share at least one Voronoi edge. If there is exactly one shared Voronoi edge, we say q (resp. p) is a single-order neighbor to p (resp. q). Otherwise, q (resp. p) is called a multiple-order neighbor of p (resp. q).

Similar to pseudo-bisectors, multiple-order neighbors are also due to low sampling density. To separate two multiple-order neighbors, we add auxiliary sites in-between them. We adopt the following strategy to ensure that the Voronoi cell of each newly added auxiliary site satisfies the closed ball property. Referring to Figure 12, both p_1 and p_2 are multiple-order neighbors to p .

—Step 1. For each Voronoi edge $\tilde{b}_i$ shared by p_1 (or p_2) and p , $i = 1, 2, 3, 4$, we find a point $x_i \in \tilde{b}_i$ that minimizes the distance from p to $\tilde{b}_i$. Then we compute $c_i \in \gamma(p, x_i)$ such that $d(p, c_i) = \varepsilon(\hat{\delta})$.

—Step 2. Each Voronoi edge $\tilde{b}_i$ has two Voronoi vertices ω_{i1} and ω_{i2} . At each ω_{ij} , $j = 1, 2$, we place a geodesic circle $\partial D(\omega_{ij}, \hat{\delta})$. There is a geodesic to p that is tangential to $\partial D(\omega_{ij}, \hat{\delta})$ and not intersecting $\tilde{b}_i$. Denote this geodesic as $\gamma(\omega_{ij})$. Let d_{ij} be the point on $\gamma(\omega_{ij})$ satisfying $d(p, d_{ij}) = \varepsilon(\hat{\delta})$.

As a summary, in the Voronoi cell $\tilde{C}(p)$, for each Voronoi edge shared by p and a multiple-order neighbor p_1 (or p_2), we add three auxiliary sites c_i, d_{i1} and d_{i2} to separate p_1 (or p_2) and p . In general, if Voronoi cell $C(p)$ shares k Voronoi edges with multiple-order neighbors, we add $3k$ auxiliary sites to separate them.

PROPOSITION 4 SEPARATING MULTIPLE-ORDER NEIGHBORS. Let $GVD(P)$, $V \subseteq P$, be a geodesic Voronoi diagram in which each Voronoi cell is a topological disk. Let $\tilde{C}(p)$ be a Voronoi cell that shares k Voronoi edges with multiple-order neighbors. Adding $3k$ auxiliary sites $\{c_i, d_{i1}, d_{i2}\}_{i=1}^k$ in the $\varepsilon(\hat{\delta})$ -neighborhood of p is topologically safe to $GVD(P)$, and is sufficient to make the updated Voronoi cells $\tilde{C}(p)$, $\{\tilde{C}(c_i), \tilde{C}(d_{i1}), \tilde{C}(d_{i2})\}_{i=1}^k$ all topological disks and satisfying the 2-cell intersection condition.

As shown in Supplementary Material, adding k auxiliary sites $\{c_i\}_{i=1}^k$ is sufficient to remove all the k multiple-order neighbors from $\tilde{C}(p)$. Moreover, we need additional $2k$ auxiliary sites $\{d_{i1}, d_{i2}\}_{i=1}^k$ to ensure that the Voronoi cells

Fig. 12. Both p_1 and p_2 are multiple-order neighbors of p . For each Voronoi edge $\tilde{b}_i$ shared by p_1 (or p_2) and p , $i = 1, 2, 3, 4$, we add 3 auxiliary sites $\{c_i, d_{i1}, d_{i2}\}_{i=1}^k$ in the $\varepsilon(\hat{\delta})$ -neighborhood of p . Then the updated Voronoi cells $\tilde{C}(p)$, $\{\tilde{C}(c_i), \tilde{C}(d_{i1}), \tilde{C}(d_{i2})\}_{i=1}^k$ are all topological disks and they satisfy the 2-cell intersection condition. See the main text for details.

Procedure 4 ensure_2-cell_intersection_condition($GVD(P)$)

- 1: Let $P' = P$.
 - 2: Sort all Voronoi edges in ascending order.
 - 3: **for** any two Voronoi edges $b_k = b_l = \{p_i, p_j\}$ with the same ordered pair **do**
 - 4: Place p_j into the multiple-order neighbor list of p_i
 - 5: Place p_i into the multiple-order neighbor list of p_j .
 - 6: **end for**
 - 7: **for** each Voronoi cell $\tilde{C}(p)$ with multiple-order neighbor(s) **do**
 - 8: **for** each shared Voronoi edge $\tilde{b}_i$ with a multiple-order neighbor **do**
 - 9: Find $x_i \in \tilde{b}_i$ that minimizes the distance from p to $\tilde{b}_i$.
 - 10: Compute the geodesic $\gamma(p, x_i)$.
 - 11: Find $c_i \in \gamma(p, x_i)$ such that $d(p, c_i) = \varepsilon(\hat{\delta})$.
 - 12: $P' = P' \cup \{c_i\}$.
 - 13: **end for**
 - 14: Locally update $GVD(P')$, $\hat{\delta}$ and $\varepsilon(\hat{\delta})$.
 - 15: Remove p from the multiple-order neighbor list of each site in p 's multiple-order neighbor list.
 - 16: **while** there is a new Voronoi cell $\tilde{C}(c_i)$ whose multiple-order neighbor list is not empty **do**
 - 17: Find two Voronoi vertices ω_{i1} and ω_{i2} of the Voronoi edge $\tilde{b}_i$.
 - 18: Compute the geodesics γ_{i1} and γ_{i2} to p tangent to $\partial D(\omega_{i1}, \hat{\delta})$ and $\partial D(\omega_{i2}, \hat{\delta})$, respectively, which do not intersect $\tilde{b}_i$.
 - 19: Find $d_{i1} \in \gamma_{i1}$ and $d_{i2} \in \gamma_{i2}$ such that $d(p, d_{i1}) = d(p, d_{i2}) = \varepsilon(\hat{\delta})$.
 - 20: $P' = P' \cup \{d_{i1}, d_{i2}\}$.
 - 21: Locally update $GVD(P')$, $\hat{\delta}$ and $\varepsilon(\hat{\delta})$.
 - 22: **end while**
 - 23: **end for**
 - 24: **return** $GVD(P')$, which satisfies the closed ball property.
-

$\{\tilde{C}(c_i), \tilde{C}(d_{i1}), \tilde{C}(d_{i2})\}_{i=1}^k$ are free of new pseudo-bisectors and multiple-order neighbors. In Procedure 4, we use an incremental sampling scheme that adds k auxiliary sites $\{c_i\}_{i=1}^k$ first. We add the additional sites d_{i1} and d_{i2} only if a Voronoi cell $\tilde{C}(c_i)$ has new multiple-order neighbors.

4.5 Computing the Dual Graph

Since the closed ball property holds everywhere for $GVD(P')$, its dual Delaunay triangulation is proper. Furthermore, each Voronoi edge has a unique dual Delaunay edge by the empty geodesic circumcircle property [Dyer et al. 2007b]. Given a Voronoi edge $\tilde{b}(p, q)$ in $GVD(P')$, we denote by $\mathcal{F}_p$ (resp. $\mathcal{F}_q$) the set of faces containing the geodesic paths from p (resp. q) to any point on $\tilde{b}(p, q)$. Then we compute the unique Delaunay edge $\gamma(p, q)$ by unfolding the faces $\mathcal{F}_p \cup \mathcal{F}_q$ to $\mathbb{R}^2$ (Procedure 5).

Procedure 5 `compute_dual_graph( $GVD(P')$ )`

- 1: **for** each Voronoi edge $\tilde{b}(p, q)$ in $GVD(P')$ **do**
 - 2: Using the windows stored at the two sides of $\tilde{b}(p, q)$, compute the face sets $\mathcal{F}_p$ and $\mathcal{F}_q$, respectively.
 - 3: Compute $\gamma(p, q)$ by unfolding the triangles in $\mathcal{F}_p \cup \mathcal{F}_q$.
 - 4: **end for**
 - 5: **for** any three Voronoi cells that meet at a Voronoi vertex **do**
 - 6: Create a geodesic triangle with the three corresponding g -edges.
 - 7: **end for**
 - 8: **return** $IDT(M)$, a proper IDT.
-

5. CORRECTNESS & COMPLEXITY ANALYSIS

THEOREM 11. *Let M be an arbitrary 2-manifold triangle mesh with n vertices. Algorithm 1 converts M into a proper IDT with $O(n)$ auxiliary sites. The algorithm has a worst-case $O(n^2 + tn \log n)$ time complexity, where t is the number of obtuse angles in the mesh.*

PROOF. First, we show that Procedure 2 `compute_gvd` takes $O(n^2 + tn \log n)$ time. The MMP algorithm [Mitchell et al. 1987] partitions each mesh edge into a set of intervals, called *windows*, which locally encode the geodesic information. For each vertex v , the MMP algorithm initializes a window w for every edge opposite to v . Such a window can be characterized by the angle associated with v that is opposite to the edge containing w . Then the MMP algorithm propagates windows across mesh faces to simulate the wavefront propagation. A window stops propagating if it hits the boundary or another window.

Our analysis is based on two key observations (see Figure 13):

- If the angle of f at v is *acute*, the window w stays inside f , i.e., it does not propagate beyond f .
- If the angle of f at v is *obtuse*, the window is propagated across f . In the worst case, a window can cross $O(n)$ faces (see Figure 14).

Since there are at most $O(n)$ obtuse angles in M , the MMP algorithm produces $O(tn)$ windows, hereby runs in $O(tn \log n)$ time. Figure 14 shows such a worst case with $t = O(n)$, implying that the bound $O(tn \log n)$ is tight. For this case, the time complexity becomes $O(n^2 \log n)$, which is a well-known result of the original MMP algorithm [Mitchell et al. 1987].

Then we show that Procedures 3, 4 and 5 all take $O(n^2)$ time. Computing $\varepsilon(\delta)$ takes $O(n^2)$ time. For each pseudo-bisector $\tilde{p}b_i$ contained in a Voronoi cell $\tilde{C}(v)$, finding $x_i \in \tilde{p}b_i$ that minimizes the distance from v to $\tilde{p}b_i$, and computing the geodesic paths $\gamma(v, x_i)$ and $\gamma'(v, x_i)$ take $O(n)$ time. Finding a_i on γ and b_i on

Fig. 13. Consider the window w associated to the edge opposite to vertex v . (a)(b) When the angle of f at v is *acute*, the window can only stay inside f , i.e., it cannot cross f . (c) When the angle of f at v is *obtuse*, the window propagates across f . The red dashed lines are the bisectors of two vertices. The region swept by window w is shaded.

Fig. 14. Complexity of Voronoi edges. (a) Each mesh vertex v_i , $i = 1, 2, \dots, n$, has two obtuse angles and each obtuse angle initializes a window that can cross $O(n)$ triangles. Therefore, the triangles on the side of the cylinder are *global*, since each of these triangles contains $O(n)$ Voronoi edges. As a result, computing $GVD(V)$ takes $O(n^2 \log n)$ time. (b) In general, if the majority of mesh faces are *local*, i.e., each edge intersects only a few Voronoi edges (see the close-up view), the GVD can be computed in $O(n)$ time.

γ' satisfying $d(v, a_i) = d(v, b_i) = \varepsilon(\delta)$ takes $O(1)$ time. Locally updating $GVD(P)$ takes $O(n)$ time. By Proposition 2, there are totally $O(n)$ pseudo-bisectors in $GVD(V)$. Thus, Procedure 3 takes $O(n^2)$ time.

Similarly, Procedure 4 adds at most $O(n)$ auxiliary sites. Since updating Voronoi cell for each new site takes $O(n)$ time, it takes $O(n^2)$ time.

In Procedure 5, computing a Delaunay edge $\gamma(p, q)$ takes $O(n)$ time, since there are at most $O(n)$ faces in the set $\mathcal{F}_p \cup \mathcal{F}_q$. Also note that there are $O(n)$ Voronoi vertices and $O(n)$ Voronoi edges in $GVD(P')$, hereby Procedure 5 takes $O(n^2)$ time.

Putting it all together, our algorithm has a worst-case time complexity $O(n^2 + tn \log n)$. $\square$

6. EXPERIMENTAL RESULTS & DISCUSSION

6.1 Performance

We implemented our algorithm in C++ and tested it on 20 synthetic and real-world models with diverse geometric and topological features. To quantitatively investigate the relation between mesh qual-

Table II. Mesh complexity and running time statistics. $|E|$: the number of edges in M ; $\#ef$: the number of edge flips by the edge-flipping algorithm; $\#as$: the number of auxiliary sites added by our algorithm; $\#sp$: the number of splitting points added by the Delaunay mesh algorithm. The 3-tuple $(\sigma_{mean}, \sigma_{max}, \sigma_{std})$ shows the mean, max and standard deviation of the anisotropy measure σ .

The running time was measured in seconds on an Intel Core i7-2600 CPU (3.40 GHz).

Model	Original mesh		Edge-flipping		IDT			DM		
	$ E $	Anisotropy σ	Time (sec)	$\#ef$	Time (sec)	$\#as$	Anisotropy σ	Time (sec)	$\#sp$	Anisotropy σ
Pant	38	(3.04, 4.48, 0.19)	0.000	14	0.000	4	(2.33, 3.34, 0.16)	0.003	12	(2.69, 4.47, 0.15)
Horse	300	(1.82, 5.30, 0.05)	0.000	58	0.000	2	(1.47, 4.15, 0.03)	0.003	107	(1.97, 18.25, 0.07)
Bunny	300	(1.71, 8.56, 0.07)	0.000	35	0.000	2	(1.47, 2.91, 0.03)	0.002	73	(1.83, 15.48, 0.07)
Kitten	300	(1.66, 16.21, 0.09)	0.000	35	0.000	2	(1.48, 5.26, 0.04)	0.002	58	(2.02, 19.61, 0.10)
CSG	357	(4.74, 33.02, 2.13)	0.006	120	0.004	2	(1.94, 4.93, 0.06)	0.009	407	(5.41, 64.13, 0.84)
Bear	400	(2.47, 40.57, 2.21)	0.003	88	0.002	2	(1.50, 3.30, 0.25)	0.004	217	(4.00, 64.80, 1.02)
Fandisk	1,081	(44.22, 177.54, 9.49)	0.015	382	0.011	0	(3.21, 0.34, 0.11)	0.125	9,530	(5.82, 92.49, 7.17)
Headst	1,714	(90.00, 330.64, 47.52)	0.060	1,693	0.001	0	(3.50, 21.03, 2.32)	0.905	93,730	(2.96, 379.12, 0.45)
Eight	3,033	(1.67, 11.80, 0.88)	0.031	717	0.016	0	(1.45, 4.81, 0.01)	0.102	8,688	(2.02, 30.59, 0.02)
Sphere	5,994	(1.67, 59.60, 2.14)	0.019	5	0.006	0	(1.67, 59.60, 2.13)	0.036	138	(1.70, 59.60, 2.12)
Lamp	12,780	(78.02, 754.60, 14.70)	0.095	3,223	0.003	0	(19.73, 5.33, 0.61)	16.679	814,943	(15.88, 102.15, 6.33)
Teapot	17,493	(1.69, 20.74, 1.66)	0.171	4,136	0.125	0	(1.03, 7.32, 0.40)	0.564	75,828	(1.95, 57.42, 2.48)
Decocube	24,000	(1.56, 13.43, 0.65)	0.188	4,227	0.140	0	(1.42, 4.02, 0.22)	0.311	45,967	(1.94, 12.23, 1.25)
Sword	29,568	(11.12, 1353.23, 3.42)	0.298	11,345	0.060	0	(9.50, 1309.48, 3.13)	1.923	224,240	(6.48, 7632.53, 2.60)
Fertility	36,920	(2.34, 17.20, 1.11)	0.609	17,127	0.561	0	(1.82, 2.76, 0.10)	1.044	112,437	(3.50, 22.37, 0.13)
Crank	60,000	(17.52, 279.54, 14.16)	1.539	66,235	1.350	0	(3.60, 3.02, 0.04)	2.210	413,768	(7.63, 12.51, 4.33)
Bunny	216,054	(1.23, 11.83, 0.19)	0.780	3,025	0.156	0	(1.15, 4.39, 0.01)	1.041	73,874	(1.29, 43.48, 0.01)
Armadillo	518,916	(1.31, 95.29, 2.39)	2.324	23,831	0.983	0	(1.30, 83.86, 2.13)	2.520	265,651	(1.52, 76.29, 1.90)
Lucy	788,721	(1.45, 34.24, 1.38)	5.179	97,418	3.960	0	(1.41, 1.22, 0.36)	7.026	924,703	(1.86, 11.17, 0.38)
Buddha	1,195,719	(1.47, 29.16, 5.22)	8.502	149,728	5.379	0	(1.26, 0.79, 1.28)	9.430	964,417	(2.24, 18.13, 1.99)

Fig. 15. Results. The original mesh edges and the Delaunay edges are colored in black and red, respectively. The figures are of high resolution, allowing close-up examination.

ity and runtime performance, we adopted the popular *anisotropy* measure [Zhong et al. 2013] $\sigma(t) = \frac{ph}{2\sqrt{3}S}$ for quality of a triangle t , where p is the half-perimeter, h is the length of its longest edge and S is the triangle area. It is easy to see that $\sigma(t) \geq 1$ and the equality holds when t is equilateral. We measure the triangulation

Fig. 16. Our algorithm empirically runs in linear time on real-world meshes. The horizontal axis shows the mesh complexity and the vertical axis is the execution time (in seconds).

quality by the mean σ_{mean} , maximum σ_{max} and standard deviation σ_{std} of σ for all triangles of M . Usually, a larger σ means the higher degree of anisotropy and the further away the mesh is from its IDT. As Table II shows, most meshes are far from their Delaunay triangulations. Take the Fertility model as an example (see Figure 15). Since 46.3% edges are non-Delaunay edges, it takes the edge-flipping algorithm 17,127 iterations to fix them. Our algorithm, in contrast, computes the IDT in a non-iterative manner and its performance is not sensitive to the number of non-Delaunay edges.

We also observe that the *theoretical* worst-case time complexity $O(n^2 \log n)$ is very pessimistic, since it happens only when each triangle contains $O(n)$ bisectors. See Figure 14(a) for a model with the worst-case time complexity. We call a triangle with $O(n)$ Voronoi edges *global*, since it indeed spans a global region on the model. We observe that on many real-world models, the majority of the triangles are not global, even though the mesh triangulation is

Fig. 17. Illustration of the edge-flipping algorithm. Consider a prism with 7 vertices. Among the four edges opposite to v_1 , v_2v_3 and v_5v_6 are locally Delaunay and the other two edges v_2v_5 and v_3v_6 are not. The edge-flipping algorithm iteratively flips the non-Delaunay edges until all edges become locally Delaunay. However, the resulting Delaunay triangulation is not proper, due to the self-loop (red) connecting v_1 and itself. Our method adds two auxiliary sites q and r in the topologically safe region of v_1 , and produce a proper IDT. The bottom row shows the 2D flattening of the region.

Fig. 18. Comparison with the edge-flipping algorithm. Due to low sampling density on the thin-handles, the IDTs produced by the edge-flipping algorithm contain self-loops (red arrows). Our method generates proper IDTs by adding auxiliary sites on the handles (green arrows).

poor (i.e., far from its Delaunay triangulation). Computational results show that on average, a mesh edge on a real-world model has only $O(1)$ Voronoi edges. See Figure 14(b). As a result, all of the four steps in our algorithm run in $O(n)$ time. Computational results confirm that our algorithm has an empirical linear time complexity (see Figure 16) and it consistently outperforms the edge-flipping algorithm in terms of execution time.

6.2 Results

As mentioned above, the edge-flipping algorithm may produce IDT with self-loops. In contrast, our method guarantees the proper IDT by adding auxiliary sites at the poorly-sampled region. See Figures 17 and 18. Theoretically, our algorithm adds $O(n)$ auxiliary sites to obtain a proper IDT. In practice, we observe that only a very

Fig. 19. Angle histograms show that IDT improves the triangulation quality.

small number of auxiliary sites are required for real-world models. This is not a surprise, since the auxiliary sites are only added on the

Algorithm 6 Constructing Proper Intrinsic Delaunay Triangulation with Uniform Samples

Input: $M = (V, E, F)$, a 2-manifold triangle mesh

Output: $IDT(M)$, a proper IDT defined on M , whose vertex set contains all vertices in V

- 1: $GVD(V) = \text{compute_gvd}(M)$ (Procedure 2)
 - 2: $GVD(P) = \text{ensure_disk_condition_2}(GVD(V))$, where $V \subseteq P$ (Procedure 7)
 - 3: $GVD(P') = \text{ensure_2-cell_intersection_condition_2}(GVD(P))$, where $P \subseteq P'$ (Procedure 8)
 - 4: $IDT(M) = \text{compute_dual_graph}(GVD(P'))$ (Procedure 5)
-

Procedure 7 $\text{ensure_disk_condition_2}(GVD(V))$

- 1: $P = V$
 - 2: Compute $\varepsilon(\hat{\delta}(GVD(V)))$ using Eqs. (S1, S6, S7) in Supplementary Material.
 - 3: **for every** $v \in V$ **do**
 - 4: **if** $C(v)$ has two or more boundaries **then**
 - 5: **for each** pseudo-bisector $\tilde{pb}_i \subset \tilde{C}(v)$ **do**
 - 6: Find $x_i \in \tilde{pb}_i$ which minimizes the distance from v to $\tilde{pb}_i$.
 - 7: Compute the Voronoi cell $\tilde{C}(x_i)$ in $GVD(V \cup \{x_i\})$.
 - 8: **if** $\tilde{C}(x_i)$ has only one connected boundary **then**
 - 9: $P = V \cup \{x_i\}$.
 - 10: Locally update $GVD(P)$, $\hat{\delta}(GVD(P))$ and $\varepsilon(\hat{\delta})$.
 - 11: **else**
 - 12: Compute two minimal geodesics $\gamma(v, x_i)$ and $\gamma'(v, x_i)$.
 - 13: Find $a_i \in \gamma$ and $b_i \in \gamma'$ such that $d(v, a_i) = d(v, b_i) = \varepsilon(\hat{\delta})$.
 - 14: $P = V \cup \{a_i\}$.
 - 15: Locally update $GVD(P)$ and $\varepsilon(\hat{\delta}(GVD(P)))$.
 - 16: **if** $C(a_i)$ has two or more disjoint boundaries **then**
 - 17: $P = P \cup \{b_i\}$.
 - 18: Locally update $GVD(P)$, $\hat{\delta}(GVD(P))$ and $\varepsilon(\hat{\delta})$.
 - 19: **end if**
 - 20: **end if**
 - 21: **end for**
 - 22: **end if**
 - 23: **end for**
 - 24: **return** $GVD(P)$, in which all Voronoi cells are topological disks.
-

sparsely-sampled regions which are not homeomorphic to a disk (e.g., the cylinder-like geometry in Figures 9 and 18). As Table II shows, although most test models are far from their Delaunay triangulations, they have a fairly good sampling density. Consequently, the resultant $GVD(V)$ automatically satisfies the closed ball density, and our algorithm does not add any new site at all. For these models, both our method and the edge-flipping algorithm produce exactly the same results. Table II and Figure 19 also show that IDTs significantly improve the triangulation quality.

6.3 An Improved Algorithm

Algorithm 1 adds new auxiliary sites near the existing sites to eliminate pseudo-bisectors and separate multiple-order neighbors. However, in practice, to obtain a fairly regular Voronoi diagram, it is

Procedure 8 $\text{ensure_2-cell_intersection_condition_2}(GVD(P))$

- 1: Let $P' = P$.
 - 2: Sort all Voronoi edges in ascending order.
 - 3: **for** any two Voronoi edges $b_k = b_l = \{p_i, p_j\}$ with the same ordered pair **do**
 - 4: Place p_j into the multiple-order neighbor list of p_i
 - 5: Place p_i into the multiple-order neighbor list of p_j .
 - 6: **end for**
 - 7: **for each** Voronoi cell $\tilde{C}(p)$ with a non-empty list of multiple-order neighbors **do**
 - 8: **for each** shared Voronoi edge $\tilde{b}_i$ with a multiple-order neighbor **do**
 - 9: Find $x_i \in \tilde{b}_i$ which minimizes the distance from p to $\tilde{b}_i$.
 - 10: Compute the Voronoi cell $\tilde{C}(x_i)$ in $GVD(V \cup \{x_i\})$.
 - 11: **if** $\tilde{C}(x_i)$ has only one connected boundary and has no multiple-order neighbor **then**
 - 12: $P' = P' \cup \{x_i\}$.
 - 13: Locally update $GVD(P')$, $\hat{\delta}$ and $\varepsilon(\hat{\delta})$.
 - 14: **else**
 - 15: Compute the geodesic $\gamma(p, x_i)$.
 - 16: Find $c_i \in \gamma(p, x_i)$ such that $d(p, c_i) = \varepsilon(\hat{\delta})$.
 - 17: $P' = P' \cup \{c_i\}$.
 - 18: Locally update $GVD(P')$, $\hat{\delta}$ and $\varepsilon(\hat{\delta})$.
 - 19: **if** $\tilde{C}(c_i)$ have multiple-order neighbor sites **then**
 - 20: Find two Voronoi vertices ω_{i_1} and ω_{i_2} of the Voronoi edge $\tilde{b}_i$.
 - 21: Compute the geodesics γ_{i_1} and γ_{i_2} to p tangent to $\partial D(\omega_{i_1}, \hat{\delta})$ and $\partial D(\omega_{i_2}, \hat{\delta})$, respectively, which do not intersect $\tilde{b}_i$.
 - 22: Find $d_{i_1} \in \gamma_{i_1}$ and $d_{i_2} \in \gamma_{i_2}$ such that $d(p, d_{i_1}) = d(p, d_{i_2}) = \varepsilon(\hat{\delta})$.
 - 23: $P' = P' \cup \{d_{i_1}, d_{i_2}\}$.
 - 24: Locally update $GVD(P')$, $\hat{\delta}$ and $\varepsilon(\hat{\delta})$.
 - 25: **end if**
 - 26: **end if**
 - 27: **end for**
 - 28: **end for**
 - 29: **return** $GVD(P')$, which satisfies the closed ball property.
-

often desired to add new sites away from the existing sites. In this section, we propose two effective strategies for eliminating pseudo-bisectors and separate multiple-order neighbors. Combining them with Procedures 3 and 4, the improved algorithm (see Algorithm 6) can output more uniform samples than those of Algorithm 1 and it can also guarantees the resulting IDTs are proper. The time complexity remains unchanged as stated in Theorem 11.

The first strategy is to add an auxiliary site on each pseudo-bisector. Let $\tilde{pb}_i$ be a pseudo-bisector in a Voronoi cell $\tilde{C}(v)$ and $x_i \in \tilde{pb}_i$ be the position that minimizes the distance from v to $\tilde{pb}_i$. It is readily seen that add an auxiliary site at x_i will remove the pseudo-bisector $\tilde{pb}_i$ from $\tilde{C}(v)$ in the updated $GVD(V \cup \{x_i\})$. However, the new Voronoi cell $\tilde{C}(x_i)$ cannot guarantee to be free of pseudo-bisectors; see a counterexample in Figure 4. In practice, we find that this strategy works well and the extremely pathological cases like the one in Figure 4 are rare. Procedure 7 combines this strategy (lines 7-10) with Procedure 3. Since locally computing $\tilde{C}(x_i)$ and examining the existence of any pseudo-bisector in

Fig. 20. Comparison with Delaunay meshes. Since the original meshes are highly anisotropic, Delaunay meshes require a large number of splitting points to ensure the local Delaunay condition. In contrast, IDT does not require any auxiliary site at all since the sampling density on the cylinder-shaped regions is high enough.

$\tilde{C}(x_i)$ take $O(n)$ time, Procedure 7 has the same time complexity as Procedure 3.

Let $\tilde{C}(p)$ be a Voronoi cell that shares k Voronoi edges with multiple-order neighbors. The second strategy is to add an auxiliary site on each of these k Voronoi edges $\{\tilde{b}_i\}_{i=1}^k$. Let $x_i \in \tilde{b}_i$ be the position that minimizes the distance from p to $\tilde{b}_i$. It is readily seen that add an auxiliary site at x_i will remove the bisector $\tilde{b}_i$ from $\tilde{C}(p)$ in the updated $GVD(P \cup \{x_i\})$. However, the new Voronoi cell $\tilde{C}(x_i)$ cannot guarantee to be free of pseudo-bisectors and multiple-order neighbors; see a counterexample in Figure 5. Akin to the first strategy, in practice, we find that the second strategy also works well and the extremely pathological cases like the one in Figure 5 are rare. Procedure 8 combines the second strategy (lines 10-13) with Procedure 4. Again, since locally computing $\tilde{C}(x_i)$ and examining the existence of any pseudo-bisector and multiple-order neighbors in $\tilde{C}(x_i)$ take $O(n)$ time, Procedure 8 has the same time complexity as Procedure 4.

See Supplementary Material for the results of Algorithm 6.

Handling degenerate cases. For constructing GVD and its dual IDT, a critical degenerate case is that more than three sites are on the same geodesic circumcircle, resulting in a nonempty intersection of more than three Voronoi cells. A common technique [Edelsbrunner and Mücke 1990] is to make a conceptual perturbation on these sites to eliminate all degeneracies. Consider four sites lying on a circumcircle: any arbitrarily small perturbation will result

in two distinct circumcircles each of which passes through three sites. However, the centers of these two circumcircles are arbitrarily close, leading to an arbitrarily small $\delta_{\max}(GVD(P))$ in Eq. (1). We handle this case by storing a local adaptive $\delta_{\max}(p)$ for each Voronoi cell $\tilde{C}(p)$, instead of using a global $\delta_{\max}(GVD(P))$. We found that together with two improved strategies in Procedures 7 and 8, this adaption works quite well in our practice. Elaborate techniques for handling all kinds of degenerate cases deserve further study in their own right and will be reported in future work.

6.4 Comparison with Delaunay Meshes

Delaunay meshes M [Dyer et al. 2007a] are manifold triangle meshes that form an intrinsic Delaunay triangulation of its vertices with respect to the piecewise flat metric of its polyhedral surface. In other words, the proper IDT associated to a Delaunay mesh is just the mesh itself. Liu et al. [2015] showed that for an arbitrary manifold triangle mesh M with n vertices, there exists a Delaunay mesh with $O(Kn)$ vertices, where K is a model-dependent constant. They also developed an efficient algorithm to construct DM in $O(nK \log K)$ time.

Both proper IDT and DM have exactly the same geometry of M and satisfy the Delaunay condition everywhere. However, they differ in representation, space complexity and time complexity.

Representation. An IDT edge is a geodesic path, which may go through many triangles. Hence, it is tedious to explicitly represent

IDT edges. Usually, one often takes the IDT as an *abstract* surface representation, i.e., ignoring the *geometry* of geodesic paths and storing only their lengths. To apply IDT to the subsequent geometry processing algorithms, one has to adapt their interfaces for the abstract representation. In contrast, Delaunay meshes are just normal triangle meshes, which can be represented by any mesh data structures, such as half edges, triangle soup, etc. Consequently, the existing geometry processing algorithms can benefit the favorable properties of Delaunay meshes without changing any code.

Space complexity. The price to pay for the flexible representation of Delaunay meshes is its high space complexity. As shown in [Liu et al. 2015], a DM has $O(Kn)$ vertices, where the constant K depends on the triangulation quality. We observe that for anisotropic meshes (especially the ones made by artists), K can be very large (even comparable with n). In contrast, we prove that a proper IDT has $O(n)$ vertices, i.e., its space complexity is independent of the mesh quality. Take the Lamp model (Figure 20) as an example. The Delaunay mesh has almost 200 times more vertices than the original mesh, whereas the IDT does not require any auxiliary sites.

Time complexity. The algorithms for constructing IDTs and DMs are also different. To construct a proper IDT, we need to compute the geodesic Voronoi diagram, which takes $O(n^2 + tn \log n)$ time. The DM construction algorithm simply flips the flippable NLD edges and splits the non-flippable NLD edges until all edges are locally. The DM construction algorithm is fairly efficient for meshes with good triangulation quality, due to its $O(nK \log K)$ time complexity. However, for anisotropic meshes such as Sword and Lamp (see Figure 20) where a large number of splitting points were added, constructing DM is highly expensive. In contrast, the performance of our method is not sensitive to the mesh quality.

In summary, both proper IDT and DM have their merits and limitations. DM is effective for triangle meshes with fairly good quality, whereas proper IDT is advantageous for anisotropic meshes.

7. CONCLUSIONS

This paper presents a new method for constructing proper intrinsic Delaunay triangulation on 2-manifold triangle meshes. Unlike the existing edge-flipping algorithm, our method is based on the dual graph of geodesic Voronoi diagram. The key idea is to add auxiliary sites to the poorly sampled regions so that the closed ball property holds everywhere, hereby the dual graph is a proper triangulation. We prove that our method produces a proper IDT by adding at most $O(n)$ sites for an n -vertex mesh. Moreover, thanks to the bounded time complexity of computing geodesic Voronoi diagrams, our method has a theoretical worst-case time complexity $O(n^2 + tn \log n)$, where t is the number of obtuse angles in the mesh. Computational results show that it empirically runs in linear time $O(n)$ on real-world models.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their detailed and valuable comments, which help us to improve the quality of the paper significantly. Special thanks go to Reviewer #1, who pointed out the case of Voronoi cells with multiple pseudo-bisectors.

REFERENCES

BOBENKO, A. I. AND SPRINGBORN, B. A. 2007. A discrete Laplace-Beltrami operator for simplicial surfaces. *Discrete & Computational Geometry* 38, 4, 740–756.

- BOISSONNAT, J., DYER, R., AND GHOSH, A. 2013. Constructing intrinsic Delaunay triangulations of submanifolds. *CoRR abs/1303.6493*.
- DYER, R., VEGTER, G., AND WINTRAECKEN, M. 2015. Riemannian simplices and triangulations. *Geometriae Dedicata* 179, 1, 91–138.
- DYER, R., ZHANG, H., AND MÖLLER, T. 2007a. Delaunay mesh construction. In *Proceedings of the Symposium on Geometry Processing (SGP 07)*. 273–282.
- DYER, R., ZHANG, H., AND MÖLLER, T. 2007b. Voronoi-Delaunay duality and Delaunay meshes. In *Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling*. 415–420.
- DYER, R., ZHANG, H., AND MÖLLER, T. 2008. Surface sampling and the intrinsic Voronoi diagram. In *Proceedings of the Symposium on Geometry Processing (SGP 08)*. 1393–1402.
- EDELSBRUNNER, H. AND MÜCKE, E. P. 1990. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics* 9, 1, 66–104.
- EDELSBRUNNER, H. AND SHAH, N. R. 1997. Triangulating topological spaces. *International Journal of Computational Geometry and Applications* 7, 4, 365–378.
- FISHER, M., SPRINGBORN, B., BOBENKO, A. I., AND SCHRÖDER, P. 2007. An algorithm for the construction of intrinsic Delaunay triangulations with applications to digital geometry processing. *Computing* 82, 2-3, 199–213.
- INDERMITTE, C., LIEBLING, T., TROYANOV, M., AND CLÉMENÇON, H. 2001. Voronoi diagrams on piecewise flat surfaces and an application to biological growth. *Theoretical Computer Science* 263, 1-2 (July).
- LIU, Y.-J. 2015. Semi-continuity of skeletons in 2-manifold and discrete voronoi approximation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 9, 1938–1944.
- LIU, Y.-J., CHEN, Z., AND TANG, K. 2011. Construction of iso-contours, bisectors, and Voronoi diagrams on triangulated surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 8, 1502–1517.
- LIU, Y.-J. AND TANG, K. 2013. The complexity of geodesic Voronoi diagrams on triangulated 2-manifold surfaces. *Information Processing Letters* 113, 4, 132–136.
- LIU, Y.-J., XU, C., FAN, D., AND HE, Y. 2015. Efficient construction and simplification of delaunay meshes. *ACM Transactions on Graphics* 34, 6, 174.
- MITCHELL, J. S., MOUNT, D. M., AND PAPADIMITRIOU, C. H. 1987. The discrete geodesic problem. *SIAM Journal on Computing* 16, 4, 647–668.
- OKABE, A., BOOTS, B., SUGIHARA, K., AND CHIU, S.-N. 2000. *Spatial Tessellations: Concept and Applications of Voronoi Diagrams*. Wiley.
- RIVIN, I. 1994. Euclidean structures on simplicial surfaces and hyperbolic volume. *Annals of Mathematics* 139, 3, 553–580.
- XU, C., LIU, Y.-J., SUN, Q., LI, J., AND HE, Y. 2014. Polyline-sourced geodesic Voronoi diagrams on triangle meshes. *Computer Graphics Forum (Pacific Graphics 2014)* 33, 7, 161–170.
- XU, C., WANG, T. Y., LIU, Y.-J., LIU, L., AND HE, Y. 2015. Fast wavefront propagation (FWP) for computing exact geodesic distances on meshes. *IEEE Transactions on Visualization and Computer Graphics* 21, 7, 822–834.
- YING, X., WANG, X., AND HE, Y. 2013. Saddle vertex graph (SVG): A novel solution to the discrete geodesic problem. *ACM Transactions on Graphics*, 170:1–12.
- ZHONG, Z., GUO, X., WANG, W., LÉVY, B., SUN, F., LIU, Y., AND MAO, W. 2013. Particle-based anisotropic surface meshing. *ACM Transactions on Graphics* 32, 4, 99:1–99:14.

Received September 2008; accepted March 2009