Pacific Graphics 2014
J. Keyser, Y. J. Kim, and P. Wonka
(Guest Editors)

Volume 33 (2014), Number 7

Polyline-sourced Geodesic Voronoi Diagrams on Triangle Meshes (Supplementary Materials)

Chunxu Xu[1], Yong-Jin Liu[1], Qian Sun[2], Jinyan Li[3] and Ying He[2]

[1]TNList, Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]School of Computer Engineering, Nanyang Technological University, Singapore
[3]Advanced Analytics Institute, University of Technology, Australia

1. Proofs of Properties in Main Paper

1.1. Proof of Property 4.2

Property 4.2 in main paper. Upon the termination of the MMP algorithm, two adjacent windows w_i and w_j have a *non-empty* co-illuminated region. Moreover, bisector $\beta(w_1, w_2)$ is in the co-illuminated region $c(w_1, w_2)$.

We prove Property 4.2 by proving the next two properties.

Property S.1 Upon the termination of the MMP algorithm, two adjacent windows w_i and w_j on edge e have a non-empty co-illuminated region.

Proof We prove this property in two cases: (1) w_i and w_j are on the same side of e and (2) w_i and w_j are on the different sides of e.
(1) w_i and w_j are on the same side of e. See Figure S1. Assume w_1 and w_2 have an empty co-illuminated region, which means that they have a 2D intersected area I in the area upon e. It is obvious that a 2D non-empty subset (denoted as I') of I must be in the triangle t adjacent to e. Consider a point p_i on edge e where w_i covers ($i = 1, 2$), we can always find such a pair of (p_1, p_2) that the geodesic paths to them intersect in I'. Denote the intersection point as x. It is easy to see that the two path: $x \to s_1 \to g_1$ and $x \to s_2 \to g_2$ both have geodesic length to x (otherwise a shorter geodesic path to p_1 or p_2 can be constructed by replacement). Then the path to p_2: $p_2 \to x \to s_1 \to g_1$ also has the geodesic distance to p_2. Since x is in a triangle face, a shortcut can be made to improve this path, a contradiction.
(2) w_i and w_j are on the different sides of e. Assume w_1 and w_2 are two such windows in an edge of triangle t. First, we note that in the MMP algorithm, if a window exists at the final stage, then its parent window must also exist at the final stage. Refer to Figure S2. Let w_3 be the parent of w_2. We claim that if projecting w_3 to the edge $\overline{AB}$ containing w_2, the projection of w_3 on $\overline{AB}$ must cover a larger portion than

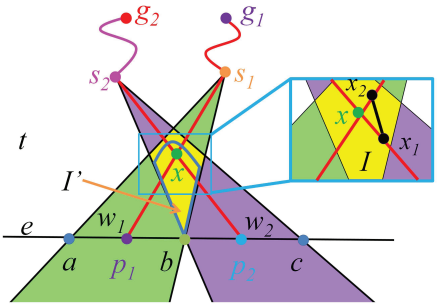


Figure S1: *Proof of Property S.1: the case that w_1 and w_2 are on the same side of e.*

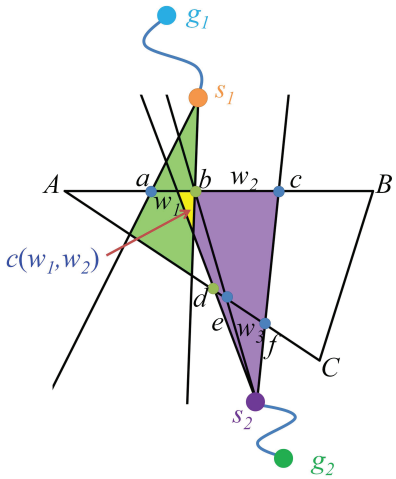
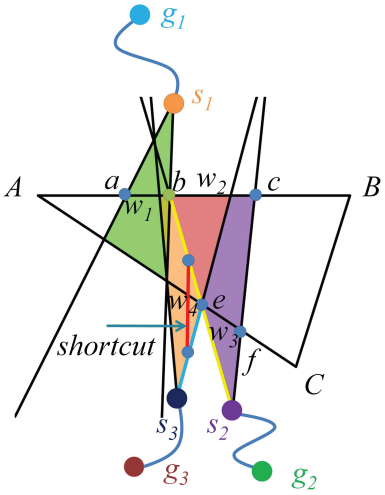
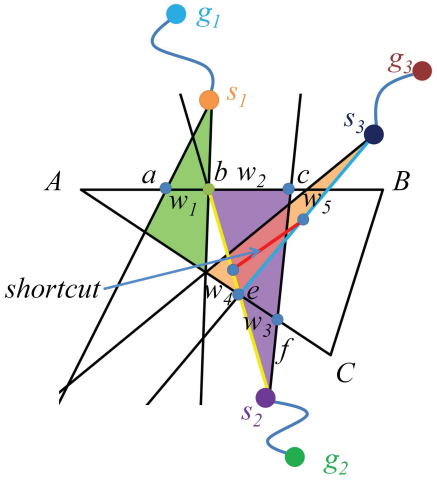


Figure S2: *Proof of Property S.1: the case that w_1 and w_2 are on the different sides of the edge.*



(a) w_3 and w_4 on the same side of the edge



(b) w_3 and w_4 on the different sides of the edge

Figure S3: *Proof of Property S.1: the case that the projection of w_3 on $\overline{AB}$ must cover the same portion of w_2, that is, the points s_2, e and b lie in the same line.*

the portion of w_2. If this claim is true, by the definition of co-illuminated region, the co-illuminated region is obviously non-empty. See also the yellow area in Figure S2 for an illustration.

The remainder of this proof is about the claim that if projecting w_3 to the edge $\overline{AB}$ containing w_2, the projection of w_3 on $\overline{AB}$ must cover a larger portion than the portion of w_2. Refer to Figure S3. If this claim is not true, then the points s_2, e and b in Figure S3 lie in the same line and there are again two cases: (a) w_3 and w_4 are on the same side of the edge, as shown in Figure S3a and (b) w_3 and w_4 are on the different sides of the edge, as shown in Figure S3b. For the case (a), there exists a shortcut (shown in red line in Figure S3a) going through window w_4 such that there is a path to the point b going through w_4 that is shorter than the shortest path to b going through w_3, a contradiction. For the case (b), let the parent of w_4 be w_5. Then similar to the case (a), there exists a shortcut (shown in red line in Figure S3b) going through window w_5 such that there is a path to the point b going through w_5 that is shorter than the shortest path to b going through w_3, a contradiction. That completes the proof. □

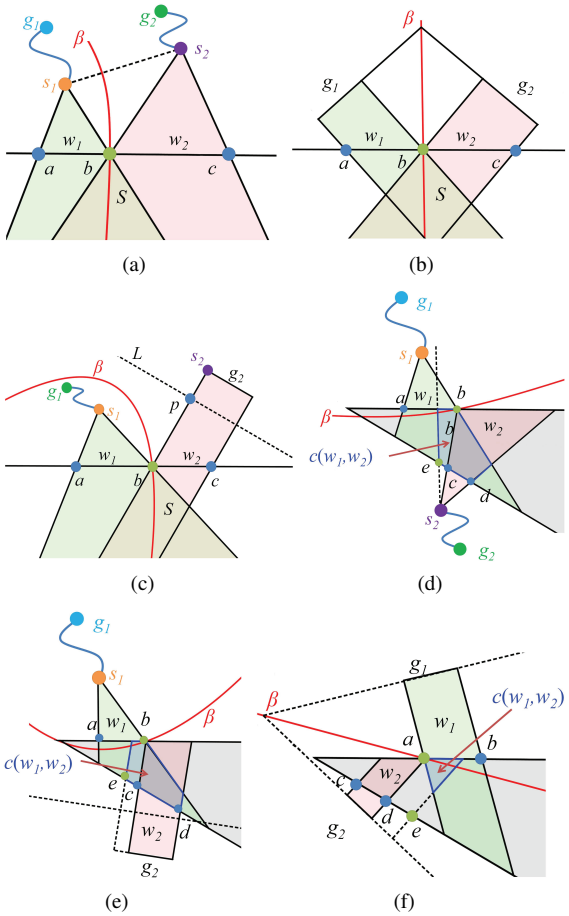


Figure S4: *Proof of Property S.2*

Property S.2 Consider two adjacent windows $w_1 = (a,b)$ and $w_2 = (b,c)$. Let g_i and s_i be the generator and pseudo-source of window w_i, $i = 1,2$. Then bisector $\beta(g_1,g_2)$ has a non-empty part in $c(w_1,w_2)$.

Proof 1) If w_1 and w_2 are projected to the same direction, see Figure S4 (a) to (c), then we denote by S the fan shaped region bordered by rays $\overrightarrow{s_1b}$ and $\overrightarrow{s_2b}$. Next we prove the property by proving that the bisector $\beta(g_1,g_2)$ has a non-empty part in S.

(1) If w_1 and w_2 are both point windows:

If $s_1 = g_1$ and $s_2 = g_2$, the bisector $\beta(g_1, g_2)$ is a line segment. Obviously it is in S. Otherwise, $\beta(g_1, g_2)$ is a hyperbolic segment and passes through the common point b, and s_1 and s_2 are the foci of $\beta(g_i, g_j)$. Therefore, both rays $\overrightarrow{s_1 b}$ and $\overrightarrow{s_2 b}$ have one intersection with $\beta(g_1, g_2)$, which is the common point b. So β has a non-empty part in S. See Figure S4(a).

(2) If w_1 and w_2 are both line windows:
By Property 4.1 in the main paper, bisector $\beta(g_1, g_2)$ is a line segment bisecting the angle formed by g_1 and g_2, so $\beta(g_1, g_2) \in S$. See Figure S4(b).

(3) If w_1 is point window and w_2 is line window:
By Property 4.1 in the main paper, the bisector $\beta(g_1, g_2)$ is a parabolic segment with s_1 as its focus and line L, parallel with g_2, as the directrix. So the ray $\overrightarrow{s_1 b}$ intersects β only once, i.e., at the common point b. And it is easy to see that $\overrightarrow{s_2 b} \perp g_2$, i.e. $\overrightarrow{s_2 b} \perp L$, then $\overrightarrow{s_2 b}$ also intersects β only once (i.e., at point b). See Figure S4(c).

2) If w_1 and w_2 are projected to opposite directions, see Figure S4(d) to (f). According to the definition of co-illuminated region, it is easy to see that there is always a part of $\beta(g_1, g_2)$ in $c(w_1, w_2)$ □

1.2. Proof of Property 4.3

Property 4.3 in main paper. Each LVD edge bisects two windows, and it does not intersect their borders.

First we prove the following property.

Property S.3 Denote the pseudo-source of window w_i as s_i, the generator of w_i as g_i, $i = 1, 2$. If $\beta(g_1, g_2)$ has a non-empty part that falls into $l(w_1) \cap l(w_2)$, then $\beta(g_1, g_2)$ does not intersect with any border of w_1 and w_2. Moreover, there will be a window (denoted as w_3) so that $\alpha(w_1, w_2, w_3) \triangleq l(w_1) \cap l(w_2) \cap l(w_3) \neq \emptyset$ and $\beta(g_3, g_1)$ (or $\beta(g_3, g_2)$) intersects $\beta(g_1, g_2)$ in $\alpha(g_1, g_2, g_3)$.

Proof First, we prove that $\beta(g_1, g_2)$ do not intersect any border of w_1 and w_2.

If w_1 is not the first/last window on its edge, see Figure S5(a), $\beta(g_1, g_2)$ has a non-empty part in $l(w_1) \cap l(w_2)$. We use contradiction to prove this. Assume that $\beta(g_1, g_2)$ intersects with w_1's boarder $\overrightarrow{s_1 a}$ at e. Thus, affected by $\overrightarrow{s_1 a}$, the next segment on the bisector will be a hyperbolic segment with the weighted point a and s_2 as foci, denoted as $\beta(a, s_2)$. Denote an arbitrary point on $\beta(a, s_2)$ as f. Since f is on $\beta(a, s_2)$, the path $f \to a \to s_1 \to g_1$ must be a geodesic path. However, a short cut can be made easily, a contradiction. Thus $\beta(s_1, s_2)$ cannot intersect $\overrightarrow{s_1 a}$, a contradiction. This contradiction can also be seen from the fact that the pseudo-source of a geodesic path can only be a saddle vertex [MMP87]. So the weighted point a in this case that is inside a triangle cannot be a pseudo-source.

If w_1 is the first/last window on its edge, see Figure S5(b), the same contradiction is held, i.e. $\beta(a, s_2)$ is generated from e. Denote an arbitrary point on $\beta(a, s_2)$ as f. If a is a convex vertex, then we can construct a short cut just as above to get a contradiction; otherwise, there will be a window w_3 with a as pseudo-source while w_1 is propagated, but w_3 may not provide a geodesic distance to f when MMP terminates (either because w_3 does not cover f or there is a shorter distance provided by another window). If w_3 does provide geodesic distance to f, then the bisector $\beta(a, s_2)$ can be treated as $\beta(g_3, g_2)$, i.e. the bisector $\beta(g_1, g_2)$ is not "cut" by $\overrightarrow{s_1 a}$ but by $\beta(g_3, g_2)$; If w_3 does not provide geodesic distance to f, i.e. the length of path $f \to a \to s_1 \to g_1$ is not geodesic distance, thus $\beta(a, s_2)$ should not exist. Whatever the case it is, $\beta(g_1, g_2)$ will not intersect $\overrightarrow{s_1 a}$.

Note that the above proof is independent on whether w_1, w_2 and w_3 are point or line windows.

Since $\beta(g_1, g_2)$ does not intersect with any border of w_1 and w_2, the reason can only be that it is "cut" by another bisector. And also, since any bisector cannot intersect with the border of its two windows, it is easy to see that a w_3 satisfying the conditions described in the property exists. □

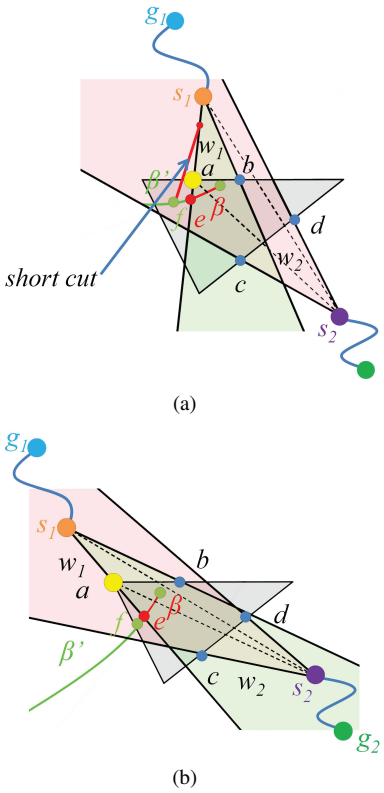


Figure S5: *Proof of Property S.3*

With Property S.3, Property 4.3 in the main paper can be proved by induction as follows: By Property S.2 and S.3, the bisector between any adjacent windows does not intersect with any border. Apply Property S.3 again, we get another pair of windows, i.e. w_1(or w_2) and w_3, with a non-empty

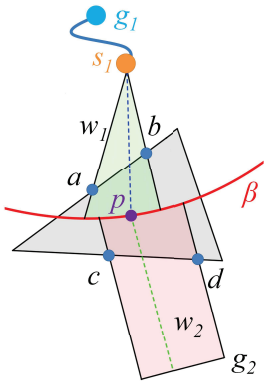


Figure S6: *Proof of Property 4.4 in main paper.*

part of its bisector in $l(w_1) \cap l(w_3)$ (or $l(w_2) \cap l(w_3)$). By iteratively applying Property S.3, the LVD is constructed, and the edges (bisectors) will never intersect with any border.

1.3. Proof of Property 4.4

Property 4.4 in main paper. The GVD restricted on a triangle t is a subset of the LVD on t, i.e., $\mathcal{G}(t) \subseteq \mathcal{L}(t)$.

Proof As Figure S6 shows, we consider an arbitrary point p on a GVD edge in the triangle t. Since p is on a bisector, there are two distinct equal-length geodesic paths to the generators g_1 and g_2. The corresponding windows containing $\gamma(p, g_1)$ and $\gamma(p, g_2)$ are w_1 and w_2, respectively. Then p is on the bisector of either an additively weighted Voronoi diagram or a line-segment Voronoi diagram generated by $s(w_1)$ and $s(w_2)$, implying that p is on the LVD edge. Therefore, any part of the GVD is a subset of the LVD. □

1.4. Proof of Property 4.5

Property 4.5 in main paper. Only two types of triangles, namely, the ones having at least one key point on its side, or the ones having a source inside, can contain GVD edges.

Proof We prove this theorem by contradiction. Assume triangle t contains a GVD edge, denoted by β, and t has neither any key point(s) on its sides nor any point- or polyline-source(s) inside it. Since there are no key points, β cannot cross the edges of t, which implies that β must be a loop that is completely inside t. By Property 4.4 in the main paper, β is a part of the LVD on t, which is a 2D Voronoi diagram. Therefore, β cannot be a loop, which is a contradiction. □

1.5. Proof of Property 5.1

Property 5.1 in main paper. On an n-face mesh with m generators, Algorithm 1 has an $O(Nn \log N)$ time complexity and an $O(Nn)$ space complexity, where $N = \max\{m, n\}$.

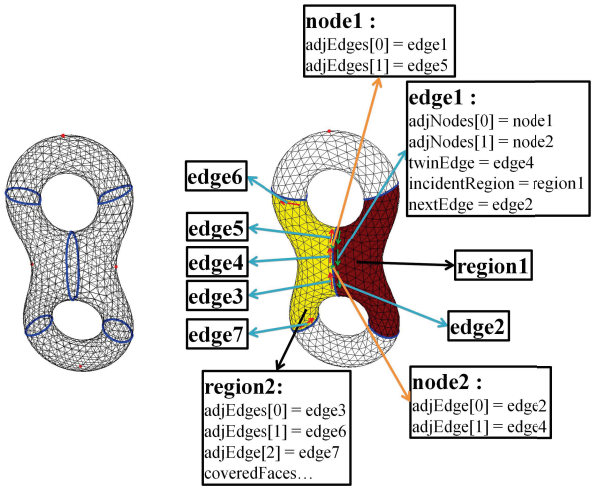


Figure S7: *Illustration of GVD structure*

Proof During the window propagation of the MMP algorithm, there are at most $O(N)$ windows at each triangle edge. Totally there are at most $O(nN)$ windows entered into the queue of window propagation. Sorting them in a priority queue takes $O(Nn \log N)$ time and $O(Nn)$ space. For each candidate triangle containing LVD, it has at most $O(N)$ windows and the plane sweep algorithm [For87] takes $O(N \log N)$ time. That completes the proof. □

2. Implementation Details

2.1. Data Structure of GVD

According to the GVD structure proposed in the main paper, we define the GVD data structure as follows. It is basically a DCEL structure ([MP78]) with slight modification:

```
struct GVD {
    Node nodes[NODESIZE];
    Edge edges[EDGESIZE];
    Region regions[REGIONSIZE];
};

//Node structure
struct Node {
    //Edges emitted from the node
    Edge* adjEdges[DEGREE];
};

struct HalfEdge {
    //Line, parabolic or hyperbolic segment
    Type type;

    //adjacent nodes
    Node* adjNodes;

    //twin half-edge
```

```
    HalfEdge* twinEdge;

    //incident Voronoi region
    Region* incidentRegion;

    //next half-edge
    HalfEdge* nextEdge;
};

struct Region {
    //Edges forming the borders of a Region
    Edge* adjEdges[EDGEAROUNDNUM];

    //triangle faces covered by the Region
    Face* coveredFaces[COVEREDFACENUM];
};
```

An illustration of the GVD structure shows in Figure S7.

2.2. Representation of a triangle mesh

The MMP algorithm proposed in [MMP87] and implemented in [SSK*05] are all based on the traditional half-edge structure. [Liu13] further discusses the correctness and efficiency of MMP algorithm on an edge based structure. In our GVD algorithm, the computation of key points is directly related to the mesh representation. Comparing with the half-edge structure in which the key points need to be obtained by "merging" the windows on each pair of half-edge, an edge-based structure will generate the key points directly without any extra calculation. Moreover, [Liu13] shows that an edge-based structure speeds up the MMP algorithm significantly both in time and space. Hence we use the edge-based structure to represent the input mesh.

2.3. Construction of LVD

There are plenty of classical algorithms to construct 2D Eucidean Voronoi diagrams with different metrics and different site types. The plane sweep algorithm (also known as *Fortune's algorithm*) is one of the optimal algorithms. Among all types of different Euclidean 2D Voronoi diagrams, two of them are important to our case. The first one is called the *additively weighted Voronoi diagram*, where the site s_i is assigned with a non-negative weight value w_i and the distance from any point p to s_i is defined as $d_E(p,s_i)+w_i$, where d_E is the Euclidean distance. The second one is defined with sites of line segments using Euclidean distance. For either of these two types of Voronoi diagrams, the plane sweep algorithm can be applied to obtain the optimal $O(nlogn)$ time bound ([For87] and [DBVKOS00]). In our case, the LVD is exactly the combination of these two types: if a pseudo-source is a point, it is a site with a non-negative weight; if a pseudo-source is a line site, it is a line source. Hence we use the plane sweep algorithm to construct the LVD structure. The details are as follows.

- We implement the event queue using a binary heap. [DBVKOS00] discussed the categories of events in both ordinary and additively weighted Voronoi diagrams: for the point sources with non-negative weights, there is one case for *site event* (event occurs when the sweepline passes through a site) and one case for *circle event* (event occurs when the sweepline passes through the lowest point of the circle defined by three sites corresponding to three consecutive parabola segments on the beach line). For the line sources, the site event also occurs when the sweepline passes the endpoints of a line segment, while the circle event have several different types according to what kind of breakpoints (a particular point on the beach line) is met when a segment on the beach line disappears. When considering non-negative weights, a point site still generates a parabolic segment on the beach line. The only difference is the order with which the corresponding site event is generated.
- A beach line is implemented as a balanced binary tree. As described in [DBVKOS00], the leaf nodes store the corresponding sites of the segments on the beach line, while internal nodes stores the breakpoints between the segments.
- LVD is stored in a DCEL structure.

Notice that when applied the case with sites of line segments, the plane sweep algorithm needs that the line segments do not intersect in their interior. However, this cannot be guaranteed as all the line segment sources are unfolded into the same plane when constructing LVD. But for a smooth (for example, a C^1 continuous) curve approximated by line segments, the line sources are usually short enough compared with its distance to the window when unfolded, so here we assume that the intersection of line segments does not exist. It is readily to see that the $O(nlogn)$ optimal time can still be held here.

2.4. Building GVD

Building GVD from LVD is accomplished in a depth-first-search (DFS) manner. Details are as follows.

Note that DFS itself takes linear time. For constructing GVD, at every time when a new pair of edges of GVD explored, a new pair of structs (objects) are added into GVD with each other's $twinEdge$ fields filled, and the $incidentRegion$ field of these two edges can also be filled in constant time. The $adjNodes$, types of the segment and $nextEdge$ field can be settled using LVD structure. On the other hand, Node structures can be built up when the corresponding Edges are filled in, so does the $adjEdges$ field of Region structure. As for the $coveredFacws$ field, it can be filled when Algorithm 1 in the main paper checks whether LVD needs to be built in line 15.

2.5. Reducing Memory Consumption

As stated in the main paper, our algorithm builds the GVD during the process of running the MMP algorithm. In fact,

building the GVD can be postponed to the time when MMP algorithm terminates. But what we consider here is the room in which the MMP can still be improved, i.e. the space cost. Building GVD inside one planar face and discarding all the windows around the boundary of that face will save the space (see the experiments in Section 6 in the main paper). Obviously, space saving usually takes more time to finish the whole algorithm. Specifically, triangles near the edges of GVD may be checked for several times before the windows around it are finally settled. Theoretically, the checking can be executed at every cn iterations of the main loop, where c is a constant number and n is the scale of mesh. See the results in Section 6 in the main paper about how different cn affects the practical efficiency of the algorithm.

References

[DBVKOS00] DE BERG M., VAN KREVELD M., OVERMARS M., SCHWARZKOPF O. C.: *Computational Geometry*. Springer, 2000.

[For87] FORTUNE S.: A sweepline algorithm for Voronoi diagrams. *Algorithmica 2*, 1-4 (1987), 153–174.

[Liu13] LIU Y.-J.: Exact geodesic metric in 2-manifold triangle meshes using edge-based data structures. *Computer-Aided Design 45*, 3 (2013), 695–704.

[MMP87] MITCHELL J. S., MOUNT D. M., PAPADIMITRIOU C. H.: The discrete geodesic problem. *SIAM Journal on Computing 16*, 4 (1987), 647–668.

[MP78] MULLER D. E., PREPARATA F. P.: Finding the intersection of two convex polyhedra. *Theoretical Computer Science 7*, 2 (1978), 217–236.

[SSK*05] SURAZHSKY V., SURAZHSKY T., KIRSANOV D., GORTLER S. J., HOPPE H.: Fast exact and approximate geodesics on meshes. In *ACM Transactions on Graphics (TOG)* (2005), vol. 24, pp. 553–560.