

# An Optimization-Driven Approach for Computing Geodesic Paths on Triangle Meshes

Bangquan Liu<sup>a,b</sup>, Shuangmin Chen<sup>a,\*</sup>, Shi-Qing Xin<sup>c</sup>, Ying He<sup>d</sup>, Zhen Liu<sup>a</sup>, Jieyu Zhao<sup>a</sup>

<sup>a</sup>Faculty of Electrical Engineering and Computer Science, Ningbo University

<sup>b</sup>College of Information Engineering, Ningbo Dahongying University

<sup>c</sup>School of Computer Science and Technology, Shandong University

<sup>d</sup>School of Computer Engineering, Nanyang Technological University

---

## Abstract

There are many application scenarios where we need to refine an initial path lying on a surface to be as short as possible. A typical way to solve this problem is to iteratively shorten one segment of the path at a time. As local approaches, they are conceptually simple and easy to implement, but they converge slowly and have poor performance on large scale models. In this paper, we develop an optimization driven approach to improve the performance of computing geodesic paths. We formulate the objective function as the total length and adopt the L-BFGS solver to minimize it. Computational results show that our method converges with super-linear rate, which significantly outperforms the existing methods. Moreover, our method is flexible to handle anisotropic metric, non-uniform density function, as well as additional user-specified constraints, such as coplanar geodesics and equally-spaced geodesic helical curves, which are challenging to the existing local methods.

## Keywords:

geodesic paths, geodesic helical curves, optimization, anisotropic metric, non-uniform density

---

## 1. Introduction

Computing geodesic distances and paths is a fundamental problem in computational geometry, and it plays an important role in a wide range of applications, including sampling [1], non-rigid registration [2], surface classification [3], geodesic Voronoi diagram [4, 5, 6], intrinsic Delaunay triangulation [7], shape segmentation [8, 9], texture mapping [10], parameterization [11, 12], image segmentation [13], and many others. Since polygonal meshes are the dominant shape representation scheme in computer graphics, solving the *discrete* geodesic problem [14] draws much attention, where the goal is to find the shortest path between two arbitrary points on a polyhedral surface. For many applications, the underlying mesh may be also associated with non-uniform density, anisotropic metric and user-specified geometric constraints.

There are two major classes of algorithms for computing discrete geodesic paths. Starting from the source point, the *global* methods propagate *discrete* wavefront across the faces in a Dijkstra-like sweep until reaching the destination. Representative works are the fast marching method [15], the Mitchell-Mount-Papadimitriou (MMP) algorithm [16], the Chen-Han (CH) algorithm [17] and their many variants [18, 19, 20, 21, 22, 23]. The global methods are able to find the *globally* shortest path, however they are computationally expensive. The *local* methods [24, 25, 26], on the other hand, adopt a completely different strategy. Starting from a given initial path connecting the two endpoints, they iteratively shorten the path until it becomes a

locally shortest geodesic. The local methods are easy to implement, but they converge slowly and can find only the local optimal solution. It is worth noting that both the existing global and local methods are designed for meshes with Euclidean metrics only, so they are not able to handle meshes with anisotropic metric and user-specified density function and/or constraints.

In this paper, we propose an optimization-driven approach to tackle the above-mentioned challenges. We formulate the objective function using the total path length in terms of the intersection points on the edge sequence containing the path, so that anisotropic metric, density function and constraints are naturally supported. Thanks to the closed-form formula of its gradient, we are able to adopt the L-BFGS solver [27] to minimize the objective function, leading to empirically super-linear convergence rate. Our method works for both open and closed geodesic paths, and it is numerically stable so that the results are insensitive to mesh tessellation and resolution. We demonstrate its efficacy on meshes with anisotropic metric, non-uniform density and user-specified constraints, such as coplanar geodesics and equally-spaced geodesic helical curves. To our knowledge, our method is the first one with all these features.

The remaining of the paper is organized as follows: Section 2 reviews the related works on discrete geodesics, followed by background knowledge in Section 3. Then Section 4 documents our method in details and Section 5 reports the experimental results and showcases a few applications. Finally, Section 6 concludes the paper and points out future directions.

---

\*Corresponding author: chenshuangmin@nbu.edu.cn (S. Chen).

## 55 2. Related Work

56 There is a large body of literature on the discrete geodesic  
57 problem. Due to space limit, we review only the most relevant  
58 work on computing geodesic paths.

59 *Global Wavefront Propagation Methods.* This type of methods  
60 is often used to compute a distance field rooted at a given source  
61 point or a globally shortest path between two given points. Most  
62 of them (e.g., [28, 16, 17, 15, 18, 29, 20, 21, 22, 23]) adopt  
63 Dijkstra’s framework to propagate discrete wavefronts. Due to  
64 the global nature, these methods are able to find the globally  
65 shortest path, but at a high computational cost. Moreover, these  
66 methods are designed for meshes with Euclidean metrics only,  
67 and it is difficult to extend them for meshes with non-uniform  
68 density function and/or anisotropic metric.

69 *PDE Methods.* Crane et al. [30] proposed a heat equation based  
70 method that initializes the source point with a unit of heat, d-  
71 iffuses it in a very short time and finally rebuilds the geodesic  
72 distance field using the gradients of the heat field. The heat  
73 method enables information reuse and outperforms the discrete  
74 wavefront propagation methods in terms of performance. It also  
75 works for point clouds and meshes with anisotropic metric [31].  
76 However, it computes only approximate solutions which are  
77 highly sensitive to mesh triangulation and resolution. Solomon  
78 et al. [32] showed that the earth mover’s distance (EMD) be-  
79 tween two delta distributions reduces to geodesic distance. Their  
80 method is able to compute a family of distances ranging from  
81 geodesic distance to bi-harmonic distance [33]. However, the  
82 high computational cost diminishes its applications to large-  
83 scale models. It is also not clear whether the PDE approaches  
84 can solve constrained geodesic problems.

85 *Graph-based Methods.* Graph-based methods solve the discrete  
86 geodesic problem by constructing an undirected graph  $G$  so that  
87 a geodesic path on the mesh  $M$  is (approximately) equal to a  
88 shortest path on  $G$ . Aleksandrov et al. [34] suggested adding  
89 a logarithmic number of Steiner points along each edge of the  
90 mesh, which are placed in a geometric progression along an  
91 edge. Then it takes  $O(mn \log(mn) + nm^2)$  time to compute a  
92  $(1 + \varepsilon)$ -approximation geodesic path, where  $m$  is the total num-  
93 ber of Steiner points. Xin et al. [35] adopted a different strat-  
94 egy by adding  $m$  (specified by the user) uniformly distributed  
95 auxiliary points on the mesh. Then they constructed a graph  
96 with  $O(m^2 + n)$  edges in  $O(mn^2 \log n)$  time. Such a graph en-  
97 ables computing geodesic distance between any pair of points  
98 (unnecessarily mesh vertices) in constant time  $O(1)$ . However,  
99 the price to pay is the quadratic space complexity of the *dense*  
100 graph, resulting poor scalability. Both saddle vertex graph and  
101 discrete geodesic graph are *sparse* graphs. Observing that most  
102 geodesic paths on real-world meshes can be partitioned into  
103 shorter segments by saddle vertices, Ying et al. [36] proposed  
104 the saddle vertex graph (SVG) to compute geodesic distances  
105 in a divide-and-conquer manner. Recently, Wang et al. [37]  
106 developed discrete geodesic graph, which enables direct error  
107 control.

108 *Local and Iterative Methods.* There are many applications where  
109 one already has an initial path and intends to refine it to a geodesic  
110 at a little computational cost. A typical technique is to iter-  
111 atively shorten the path by local perturbation like shrinking a  
112 rubber band [24, 25, 26]. These approaches are easy to imple-  
113 ment, however they have only linear convergence rate, and their  
114 performance is highly sensitive to initialization. In this paper,  
115 we propose a better iterative scheme with an explicit objective  
116 function whose gradient can be evaluated in closed-form formu-  
117 la. This allows us to adopt the efficient L-BFGS solver to  
118 minimize the function. This seemingly minor change brings at  
119 least two benefits. On one hand, it significantly improves the  
120 performance due to its superlinear convergence rate. On the  
121 other hand, the new algorithm is purely optimization driven,  
122 hereby it naturally supports anisotropic metric, user-specified  
123 density function and constraints.

## 124 3. Preliminary

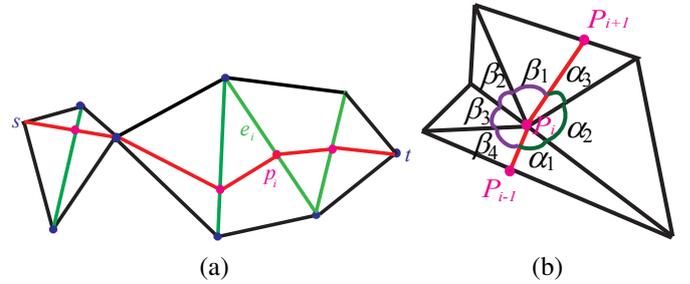


Figure 1: (a) A discrete path can be encoded using the face sequence containing it. (b) A path  $(P_{i-1}, P_i, P_{i+1})$  passes through vertex  $P_i$  and it splits the cone angle at  $P_i$  into two parts. To be an *exact* geodesic path, both angles  $\sum_{i=1}^3 \alpha_i$  and  $\sum_{j=1}^4 \beta_j$  are greater than or equal to  $\pi$ . In contrast, our algorithm computes an *approximate* geodesic path by relaxing such a constraint.

125 Denote by  $M = (V, E, F)$  a connected, manifold triangle  
126 mesh, where  $V, E, F$  are the vertex, edge and face sets respec-  
127 tively. A vertex is called spherical, Euclidean or saddle if its  
128 cone angle is less than, equal to, or greater than  $2\pi$ . Given two  
129 points  $p, q \in M$ , the geodesic path between  $p$  and  $q$  is denot-  
130 ed by  $\gamma(p, q)$ . Since geodesic paths are not unique in general,  
131 unless otherwise specified,  $\gamma(p, q)$  refers to the global shortest  
132 geodesic path in this paper. Mitchell et al. [16] showed that a  
133 geodesic path on a mesh is an alternating sequence of vertices  
134 and (possibly empty) edge sequences such that the unfolded im-  
135 age of the path along any edge sequence is a straight line seg-  
136 ment and the angle of  $\gamma$  passing through a vertex is greater than  
137 or equal to  $\pi$ , as shown in Figure 1(a).

138 The existing local geodesic methods [25, 24, 26, 38] are  
139 mainly designed based on this property. They generally require  
140 two common operations, shortening the path and updating the  
141 edge sequence containing it. Commonly used path shorten-  
142 ing techniques are graph refinement [25], Laplacian smooth-  
143 ing [24] and edge sequence unfolding [26, 38], in which the  
144 angle checking rule is strictly enforced at each iteration, i.e.,  
145 both of the angles split by the path are greater than or equal  
146 to  $\pi$ . However, they lack a clear objective function defined on

147 the whole path, which is crucial to optimization. When the base  
 148 mesh is equipped with anisotropic metric and non-uniform den-  
 149 sity function or user-specified geometric constraints, the desir-  
 150 able path does not follow the angle checking rule and therefore  
 151 the existing local methods cannot deal with these complicated  
 152 cases. This motivates us to develop an optimization driven ap-  
 153 proach to refine the intimal path without any angle checking  
 154 operations.

## 155 4. Algorithm

### 156 4.1. Formulation

157 Let  $\mathcal{S}$  be a triangulated polyhedral surface in  $\mathcal{R}^3$ , defined by  
 158 a set of vertices, edges, and faces. Given two points  $s, t \in \mathcal{S}$ ,  
 159 there are infinitely many paths between them. Our task is to  
 160 refine an initial path to an as-short-as-possible path that meets  
 161 given requirements. Let's assume the base mesh have a uniform  
 162 density and identity metric at this moment. We will extend  
 163 the algorithmic framework to non-uniform density setting and  
 164 anisotropic metric in Section 4.3.

As Figure 1 shows, the initial path (shown in red) through  
 the face sequence  $\Gamma$  can be described by  $(s, p_1, p_2, \dots, p_k, t)$ ,  
 where  $p_i$  is the intersection point between the path and the  $i$ -  
 th edge  $e_i \triangleq (v_{e_i}^{(1)}, v_{e_i}^{(2)})$  in the face sequence. Then  $p_i$  can be  
 represented by a scalar  $\lambda_i \in [0, 1]$  so that  $p_i = (1 - \lambda_i)v_{e_i}^{(1)} + \lambda_i v_{e_i}^{(2)}$ .  
 The total path length is

$$L(\Gamma; \lambda_1, \lambda_2, \dots, \lambda_k) = \|\vec{s p_1}\| + \sum_{i=1}^{k-1} \|\vec{p_i p_{i+1}}\| + \|\vec{p_k t}\|. \quad (1)$$

It is easy to compute the partial derivatives

$$\frac{\partial L}{\partial \lambda_i} = \begin{cases} (v_1^{(2)} - v_1^{(1)}, \frac{\vec{s p_1}}{\|\vec{s p_1}\|} - \frac{\vec{p_1 p_2}}{\|\vec{p_1 p_2}\|}), & i = 1, \\ (v_i^{(2)} - v_i^{(1)}, \frac{\vec{p_{i-1} p_i}}{\|\vec{p_{i-1} p_i}\|} - \frac{\vec{p_i p_{i+1}}}{\|\vec{p_i p_{i+1}}\|}), & 1 < i < k, \\ (v_k^{(2)} - v_k^{(1)}, \frac{\vec{p_{k-1} p_k}}{\|\vec{p_{k-1} p_k}\|} - \frac{\vec{p_k t}}{\|\vec{p_k t}\|}), & i = k. \end{cases} \quad (2)$$

Suppose that the user-specified constraints can be represented  
 by  $H(\Gamma; \lambda_1, \lambda_2, \dots, \lambda_k) = 0$ . To this end, our task is to find a  
 face sequence  $\Gamma$ , as well as a collection of numbers  $\lambda_1, \lambda_2, \dots, \lambda_k$ ,  
 so as to

$$\text{minimize } L(\Gamma; \lambda_1, \lambda_2, \dots, \lambda_k) + M \times H^2(\Gamma; \lambda_1, \lambda_2, \dots, \lambda_k)$$

subject to

$$0 \leq \lambda_i \leq 1, 1 \leq i \leq k,$$

165 where  $M$  is a sufficiently large coefficient. In general, a large  
 166 penalty parameter  $M$  leads to solutions satisfying the constraint,  
 167 but it often increases the fluctuation of the objective function.  
 168 In our setting, we empirically set  $M$  so that the penalty term is  
 169 about 10 times the total length.

### 170 4.2. Implementation

171 Different from general optimization problems, the as-short-  
 172 as-possible path problem needs to update the face sequence in  
 173 the optimization process. Our basic idea is that during each

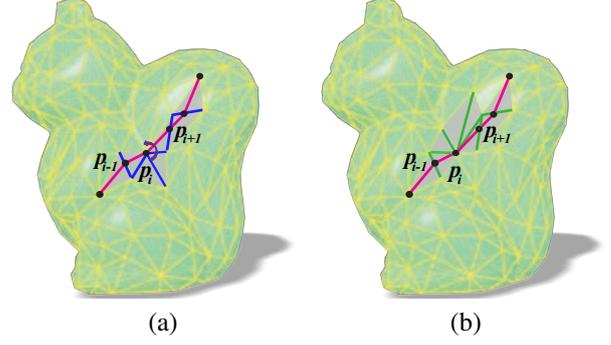


Figure 3: If the current path passes through one of the vertices, say,  $p_i$ , then we  
 update the face sequence from (a) to (b).

174 iteration we perform the following operations: (1) compute the  
 175 value and the gradients of  $L + M \times H^2$ , (2) update  $\lambda_1, \lambda_2, \dots, \lambda_k$   
 176 according to the direction and the step size recommended by L-  
 177 BFGS, (3) pull each  $\lambda_i$  to  $[0, 1]$ , and (4) update the current face  
 178 sequence where  $\lambda_i = 0$  or  $\lambda_i = 1$ .

179 We use Figure 2 to demonstrate a refining process on a 2D  
 180 mesh. It can be seen from the plots of path length and its gradi-  
 181 ent norm that our algorithm has a desirable convergence rate  
 182 and greatly outperforms the existing iterative algorithm pro-  
 183 posed by [24]; See the length and gradient-norm decreasing  
 184 plots in this figure. We will evaluate its performance in detail  
 185 in the next section. The pseudo-code is shown in Algorithm 1.

186 As one of the key routines, we have to update the face se-  
 187 quence when  $\lambda_i$  reaches 0 or 1. Taking Figure 3 as an example,  
 188 the current path pass through  $p_i$  that is a mesh vertex.  $p_i$  is  
 189 on the left side of the current triangle strip (whose boundary  
 190 consists of two sequences of vertices). We traverse the face se-  
 191 quence from beginning to end and change the face sequence  
 192 such that  $p_i$  is on the right side of the new face sequence.

**Input:** A triangle Mesh  $\mathcal{S}$ ; two points  $s, t \in \mathcal{S}$  and an  
 initial path between them; A error tolerance  $\epsilon$ .

**Output:** A geodesic path between  $s$  and  $t$ .

1 **while** the length difference between two successive  
 iterations is larger than  $\epsilon$  **do**

- 2     Compute the length of  $L(\Gamma) + M \times H^2(\Gamma)$ ;
- 3     Compute the gradient  $\nabla(L(\Gamma) + M \times H^2(\Gamma))$ ;
- 4     Get the resulting path;
- 5     Collect the vertices  $\{v_i\}$  on the path;
- 6     Traverse and update the face sequence such that each  
        $v \in \{v_i\}$  is on the other side of the new face sequence.

7 **end**

**Algorithm 1:** A optimization based method for computing an  
 as-short-as-possible path.

193 **Remark:** Our algorithm has two differences from the con-  
 194 ventional iterative scheme. First, the conventional iterative al-  
 195 gorithm does not update face sequence until the path constrained  
 196 on the current face sequence has been found, while our algo-  
 197 rithm performs face sequence updating when  $\lambda_i = 0$  or  $\lambda_i = 1$ .  
 198 Second, when updating the face sequence, the conventional iter-  
 199 ative algorithm needs to check the angles passing through each

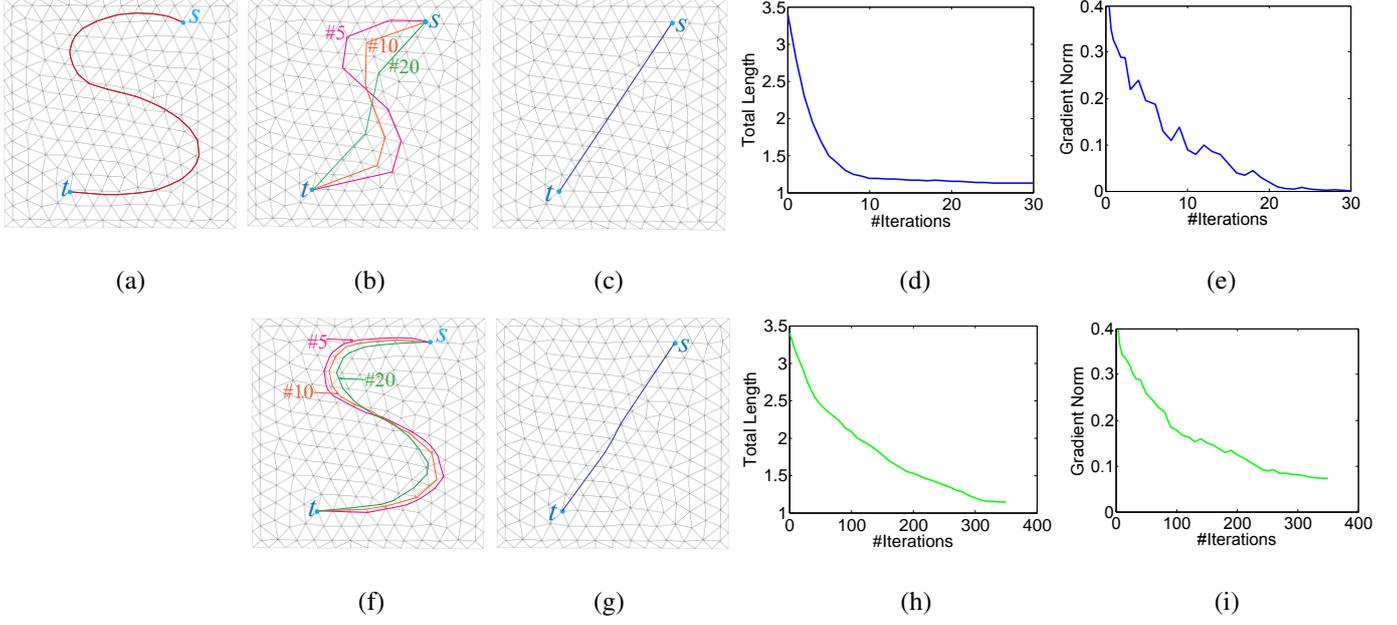


Figure 2: An example on a 2D mesh. (a) shows the initial path. (b) Three intermediate results for iteration 5, 10, and 20. (c) Our algorithm terminates after 30 iterations, with the gradient norm less than  $10^{-6}$ . The resulting path has a relative error less than  $10^{-6}$ . (d) and (e) show the plots of objective function and its gradient norm. (f) shows three intermediate results of Martínez et al.’s algorithm [24]. (g) After 350 iterations, their algorithm produces the path with relative error 0.3%. As shown in (h) and (i), their algorithm has rather low convergence rate comparing to ours.

200 vertex to guarantee that at the conclusion of the algorithm, the  
 201 angles on both sides are greater than or equal to  $\pi$ , as Figure 1(b)  
 202 shows. However, our goal is to find an as-short-as-possible path and thus our algorithm does not include such a step of angle  
 203 checking.  
 204

### 205 4.3. Meshes with Non-Uniform Density and/or Anisotropic Metric

206 First, we will discuss the case where the base mesh has a non-uniform density function defined on the surface. Typically, each polygonal face  $f_i$  has a constant density value  $\rho(f_i)$ . In this case, the total path length is given by

$$L(\Gamma; \lambda_1, \lambda_2, \dots, \lambda_k) = \rho(\overrightarrow{sp_1}) \|\overrightarrow{sp_1}\| + \sum_{i=1}^{k-1} \rho(\overrightarrow{p_i p_{i+1}}) \|\overrightarrow{p_i p_{i+1}}\| + \rho(\overrightarrow{p_k t}) \|\overrightarrow{p_k t}\|, \quad (3)$$

whose gradient with regard to  $\lambda_i$  is

$$\frac{\partial L}{\partial \lambda_i} = \begin{cases} (v_1^{(2)} - v_1^{(1)}, \rho(\overrightarrow{sp_1}) \frac{\overrightarrow{sp_1}}{\|\overrightarrow{sp_1}\|} - \rho(\overrightarrow{p_1 p_2}) \frac{\overrightarrow{p_1 p_2}}{\|\overrightarrow{p_1 p_2}\|}), & i = 1 \\ (v_i^{(2)} - v_i^{(1)}, \rho(\overrightarrow{p_{i-1} p_i}) \frac{\overrightarrow{p_{i-1} p_i}}{\|\overrightarrow{p_{i-1} p_i}\|} - \rho(\overrightarrow{p_i p_{i+1}}) \frac{\overrightarrow{p_i p_{i+1}}}{\|\overrightarrow{p_i p_{i+1}}\|}), & 1 < i < k \\ (v_k^{(2)} - v_k^{(1)}, \rho(\overrightarrow{p_{k-1} p_k}) \frac{\overrightarrow{p_{k-1} p_k}}{\|\overrightarrow{p_{k-1} p_k}\|} - \rho(\overrightarrow{p_k t}) \frac{\overrightarrow{p_k t}}{\|\overrightarrow{p_k t}\|}), & i = k, \end{cases} \quad (4)$$

207 where  $\rho(\overrightarrow{AB})$  denotes the density in the face containing the line  
 208 segment  $\overrightarrow{AB}$ . It’s easy to know that if the density is uniform  
 209 on the surface, then the above formulation degenerates into E-  
 210 q. (1) and Eq. (2). Otherwise, the path tends to go through the  
 211 area with a low density. As Figure 4 shows, we locally set a  
 212 Gaussian density function to Bimba (400K faces), Horse (40K

213 faces) and Buddha (600K faces) respectively to see how the  
 214 density setting influences the geodesic paths. It can be seen that  
 215 the density-weighted geodesics (shown in blue) tend to pass  
 through the area with low density.

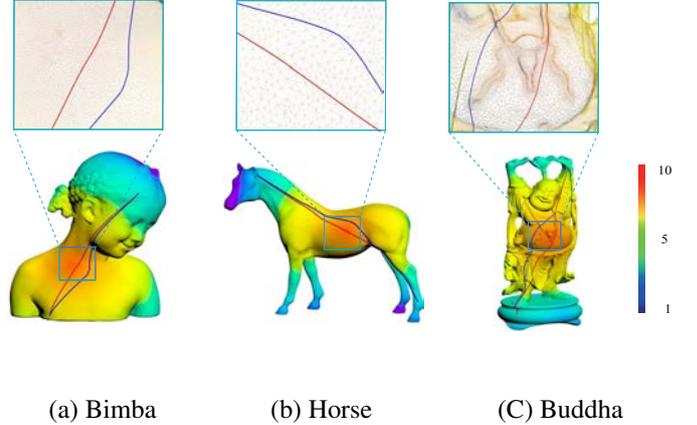


Figure 4: We locally set a Gaussian density function to Bimba, Horse and Buddha respectively and observe how the density setting influences the geodesic paths. The red paths are geodesics assuming the density is uniform on the surface, while the blue paths are geodesics in the non-uniform density setting.

When the input mesh carries an anisotropic metric, we can assume that each face is associated with two orthogonal vectors  $(\vec{\tau}_1, \vec{\tau}_2)$  in the discrete setting. In real computation, we assign each face with a  $2 \times 2$  matrix to denote the anisotropic metric operator. Let  $T(f_i)$  be the matrix on the face  $f_i$ . The total path length is given by

$$L(\lambda_1, \lambda_2, \dots, \lambda_k) = \|T(\overrightarrow{sp_1})\| + \sum_{i=1}^{k-1} \|T(\overrightarrow{p_i p_{i+1}})\| + \|T(\overrightarrow{p_k t})\|, \quad (5)$$

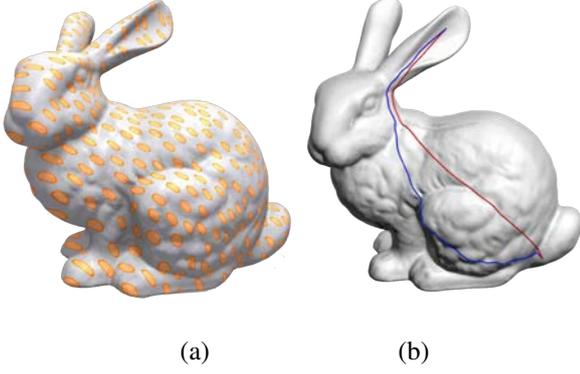


Figure 5: Anisotropic/Isotropic geodesic paths. The red path in (b) is a geodesic under the isotropic and uniform metric, while the blue path is computed under the anisotropic metric shown in (a).

whose gradient with regard to  $\lambda_i$  is

$$\frac{\partial L}{\partial \lambda_i} = \begin{cases} (v_1^{(2)} - v_1^{(1)}, \frac{T(\overrightarrow{sp_1})}{\|T(\overrightarrow{sp_1})\|} - \frac{T(\overrightarrow{p_1p_2})}{\|T(\overrightarrow{p_1p_2})\|}), & i = 1 \\ (v_i^{(2)} - v_i^{(1)}, \frac{T(\overrightarrow{p_{i-1}p_i})}{\|T(\overrightarrow{p_{i-1}p_i})\|} - \frac{T(\overrightarrow{p_i p_{i+1}})}{\|T(\overrightarrow{p_i p_{i+1}})\|}), & 1 < i < k, \\ (v_k^{(2)} - v_k^{(1)}, \frac{T(\overrightarrow{p_{k-1}p_k})}{\|T(\overrightarrow{p_{k-1}p_k})\|} - \frac{T(\overrightarrow{p_k t})}{\|T(\overrightarrow{p_k t})\|}), & i = k \end{cases} \quad (6)$$

where  $T(\overrightarrow{AB})$  denotes the  $2 \times 2$  matrix encoding the anisotropic metric in the face containing the line segment  $\overrightarrow{AB}$ . As Figure 5 shows, the left-side figure shows the anisotropic metric on the surface, while the right-side figure shows two geodesics under the isotropic/anisotropic metrics. To sum up, even if the base mesh has a non-uniform density setting or an anisotropic metric, we can still use the same optimization based algorithmic framework to compute the as-short-as-possible path that meets the user-specified constraints.

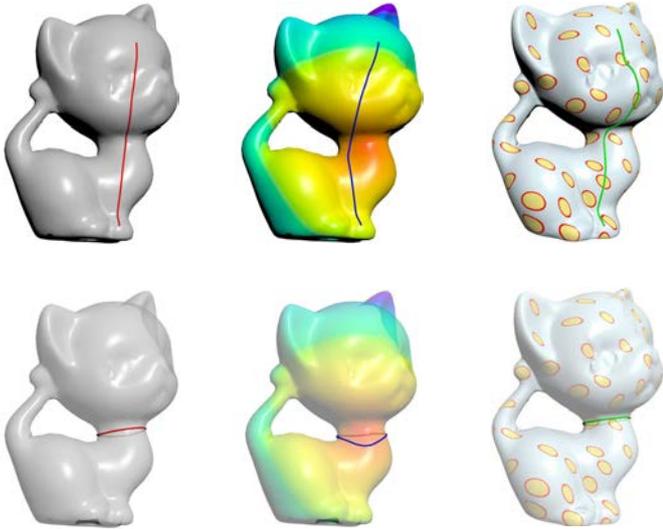


Figure 6: Computing open/closed geodesic using our algorithm. The model in the left column adopts isotropic metric, whereas the middle and right columns show the models with non-uniform density and anisotropic metric respectively.

## 5. Results & Applications

We implemented our algorithm in C++ and OpenMP, and conducted the experiments on a PC with a 3.07GHz Intel(R) Core(TM) i7 CPU and 6 GB memory.

### 5.1. Qualitative Comparison

First, we evaluated its performance and compared it to five representative methods:

- The geodesic algorithm in this paper, where the initial path is given by FMM [15];
- The fast marching method (FMM) [15];
- The improved fast marching method (IFMM) [26];
- The iterative method by Martínez et al. [24], where the initial path is given by FMM [15].
- The improved Chen & Han's (ICH) algorithm [20];
- The fast wavefront propagation (FWP) geodesic algorithm [22].

In Table 1, we show qualitative comparison of the above-mentioned five representative methods. In Figure 6, we show some examples of open/closed geodesics on the Kitten model, where the left column doesn't assume any density setting or anisotropic metric, the middle column assumes a non-uniform density function, the right column assumes an anisotropic metric setting. To summarize, our algorithm is able to compute open/closed geodesics, handle non-uniform density setting and anisotropic metrics, and support additional geometric constraints. These merits distinguish our algorithm from the existing geodesic approaches.

It's worth noting that the iterative method by Martínez et al. [24] cannot support non-uniform density setting or anisotropic metric since it has to perform an angle checking step as Figure 1(b) shows.

### 5.2. Quantitative Comparison

Now let's assume that the input doesn't have non-uniform density or anisotropic metric and observe the performance contrast. For comparison purpose, we set the termination condition to be  $|L - L^*|/L^* < 0.1\%$ , where  $L^*$  is the exact geodesic distance. At the same time, we randomly sample 10K pairs of vertices to take down the average computation time.

We use the Bunny model as the test model, which is meshed into various resolutions ranging from 20K to 100K faces. Table 2 shows the timing statistics, from which we can clearly see an performance advantage compared with the existing algorithms, whether local scheme [24] or global ones [20, 15]. See Figure 7(a). We must point out that the approach by Martínez et al. [24] only has linear convergence rate since it is not guided by gradients. Our algorithm takes advantage of gradients and benefits from the L-BFGS solver. For the Bunny model with 100K faces, our algorithm takes 1.15 seconds averagely to compute a geodesic path while the iterative algorithm in [24] requires

Table 1: Features of geodesic algorithms

Method	Open geodesic	Geodesic loop	Density function	Anisotropy metric	Geometric constraints
Ours	✓	✓	✓	✓	✓
FMM	✓	×	✓	✓	×
IFMM	✓	×	×	×	×
ICH	✓	×	×	×	×
FWP	✓	×	×	×	×
Martínez et al.	✓	✓	×	×	×

Table 2: Performance of representative algorithms on the Bunny model in various resolutions

#Faces	Ours		IFMM [26]		Our+IFMM		Xin-Wang [20]		FWP [22]		Martínez et al. [24]	
	Time (s)	$ L - L^* /L^*$	Time (s)	$ L - L^* /L^*$	Time (s)	$ L - L^* /L^*$	Time (s)	$ L - L^* /L^*$	Time (s)	$ L - L^* /L^*$	Time (s)	$ L - L^* /L^*$
20K	0.18	0.001	0.23	1.217	0.41	0.001	0.65	-	0.35	-	10.29	0.001
40K	0.25		0.43	0.958	0.68		1.93		0.62		18.48	
60K	0.38		0.75	0.786	1.13		4.17		1.78		34.46	
80K	0.65		1.04	0.518	1.69		7.24		2.89		56.63	
100K	0.98		1.63	0.485	2.66		14.11		5.16		83.53	
150K	1.89		3.15	0.366	5.04		23.87		8.04		108.86	
200K	3.21		6.56	0.187	9.77		37.11		14.61		160.34	
300K	4.96		11.12	0.089	16.08		58.89		21.78		246.65	

275 85 seconds in average. We show the length decreasing plots in  
 276 Figure 7(b). The significant contrast shows that our algorithm  
 277 has super-linear convergence rate. It’s worth noting that their  
 278 iterative algorithm converges very slowly especially when the  
 279 path is close to the exact geodesic. In spite of this, we must  
 280 point out that both of them depends on initialization and may  
 281 converge into a geodesic path that is different from the globally  
 282 shortest path.

### 283 5.3. Applications

284 In this section, we shall exhibit its uses to three typical ap-  
 285 plication occasions including (1) closed geodesics, (2) coplanar  
 286 geodesics on surfaces and (3) equally-spaced geodesic helix  
 287 curves.

#### 288 5.3.1. Geodesic Loops

289 Geodesic loops [38, 39], or closed geodesic curves, are use-  
 290 ful in mesh cutting, girth estimation, and topology analysis. A  
 291 closed curve is said to be a geodesic loop if every subpath is an  
 292 open geodesic. Note that for most problems, “stable” geodesic  
 293 loops robust to small perturbation are of our interest. Xin et  
 294 al. [38] proposed a shortening technique to compute geodesic  
 295 loops but their method has to repeatedly unfold a long face se-  
 296 quence, which is a tedious and time-consuming step. Wu and  
 297 Tai [39] suggested finding the level set where the geodesic cur-  
 298 vature is zero. However, their method doesn’t have a desirable  
 299 performance according to their timing statistics. Obviously our  
 300 algorithm can be naturally extended to compute geodesic loops  
 301 without any modification as long as it is initialized by a closed  
 302 curve, see in Figure 8. Compared with the existing methods,  
 303 our algorithm has a better performance. For example, for the  
 304 horse model with 20K faces, it only takes 0.19 seconds in aver-  
 305 age to compute a geodesic loop, which shows that our algorithm  
 306 can be used to interactive mesh cutting. Besides, we must point

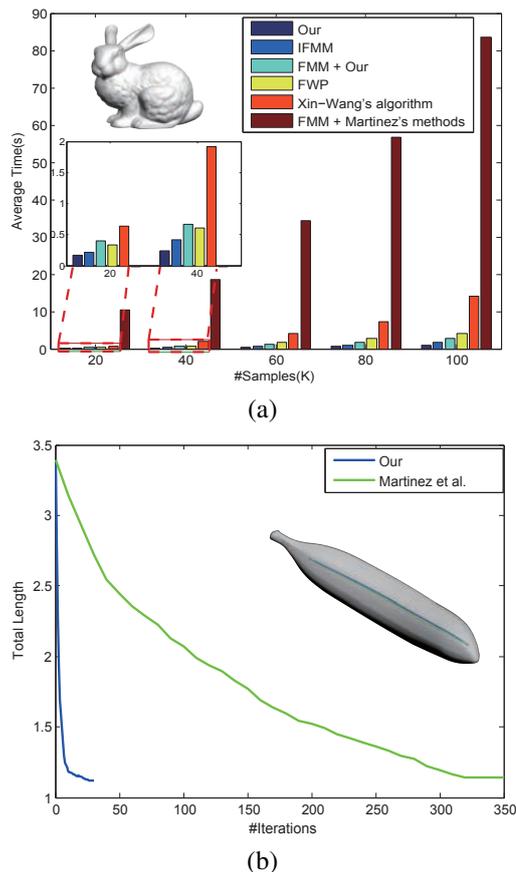


Figure 7: Performance comparison. Our algorithm has a performance advantage compared with the existing algorithms (a), which is due to the super-linear convergence rate (b). Note that the iterative approach [24] only has linear convergence rate since it is not guided by gradients.

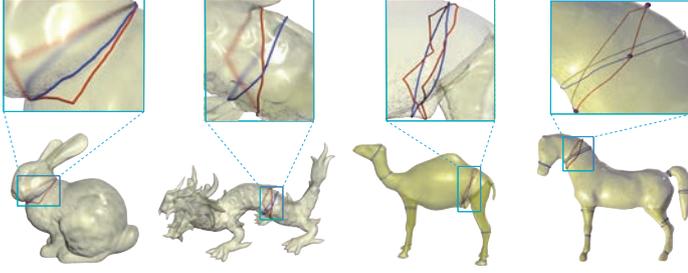


Figure 8: Our algorithm is able to compute geodesic loops.

307 out even if the base mesh has a background density function or  
 308 anisotropic metric, our algorithm still works well, which is a  
 309 desirable feature in many practical applications.

### 310 5.3.2. Co-Planar Geodesics

In the computer graphic community, there are many problems that need to find an as-short-as-possible path that meet the user-specified constraints. For example, given an initial open/closed curve, can we find a coplanar curve that is as short as possible? That is to say, we need to minimize the total length

$$L(\lambda_1, \lambda_2, \dots, \lambda_k) = \|\vec{s}p_1\| + \sum_{i=1}^{k-1} \|\vec{p}_i p_{i+1}\| + \|\vec{p}_k t\|, \quad (7)$$

subject to

$$\begin{cases} \vec{n} \cdot p_1 - d = 0, \\ \vec{n} \cdot p_2 - d = 0, \\ \vdots \\ \vec{n} \cdot p_k - d = 0. \end{cases} \quad (8)$$

311 where  $d$  is the signed distance from the origin to the plane,  $k$   
 312 is the number of geodesic points, and  $\vec{n}$  is the plane normal.

313 Figure 9 gave a group of examples on co-planar geodesics.

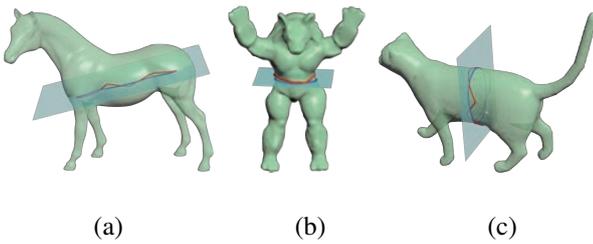


Figure 9: Coplanar geodesics. In (a), (b) and (c), the blue paths are the resulting coplanar geodesics refined from the initial red paths.

### 314 5.3.3. Geodesic Helix Curves

Another example is the geodesic winding problem [40, 41], where a family of equally spaced geodesic curves needs to be spread out on the surface of an input 3D object, which is very useful in the manufacturing field. However, it's highly non-trivial to guarantee that the geodesic curves are uniformly distributed on the surface. Xin et al. [41] pointed out that a geodesic helix curve tends to pass through the narrow area. Therefore, we want to find a special density setting such that the curve

length gets punished when passing through the narrow area. We observe that such a density setting is desirable: for each point  $p$  on the surface, we set the density  $\rho(p)$  to be in inverse proportion to its "girth", denoted by  $g(p)$ , that is defined to be the length of the shortest  $p$ -constrained geodesic loop [42]. Mathematically, the weighted path length is

$$L = \int_0^1 \frac{1}{g(p)} ds.$$

315 In implementation, we use the method proposed in [42] to pre-  
 316 compute the girth function, and then generate an N-winding  
 317 initial path on the surface. Finally we use the algorithm in this  
 318 paper to refine it to be a geodesic. It's worth noting that Xin  
 319 et al.'s algorithm [41] does not support user-specified density  
 320 function, hereby their result is far from being equally spaced.  
 321 In sharp contrast, our algorithm can naturally deal with non-  
 322 uniform density function, and produce equally spaced helical  
 curve. See Figure 10.

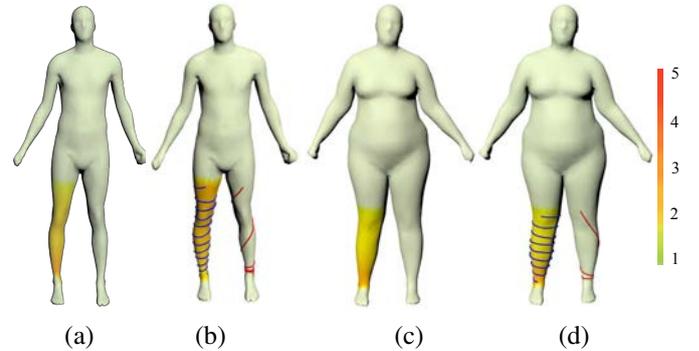


Figure 10: Equally spaced geodesic helical curves on synthetic and real-world models. The density functions (a) and (c) are visualized using color map, where the warm and cold colors denote high and low densities respectively. In (b) and (d), the red curves are the quasi-helices computed by Xin et al. [41] . while the blue curves are computed by our algorithm.

323

## 324 6. Conclusions and Future Works

325 In this paper, we propose an optimization based approach  
 326 that is different from conventional algorithmic schemes. The  
 327 distinguished feature of the new algorithmic framework is that  
 328 it supports non-uniform density setting, anisotropy metric and  
 329 additional geometric constraints at the same time. For example,  
 330 the new algorithm enables us to compute co-planar geodesics-  
 331 or equally-spaced geodesic helix curves. Equipped with an  
 332 L-BFGS solver, our algorithm exhibits a high computational  
 333 performance. We believe that it has a great potential for many  
 334 computer graphics occasions especially when one wants to find  
 335 an as-short-as-possible path while meeting some specific geo-  
 336 metric requirements. In the future, we will keep improving the  
 337 performance by advanced parallelization techniques. We will  
 338 also explore more applications in visual computing.

### 339 Acknowledgement

340 We are grateful to the editors and anonymous reviewers  
 341 for their insightful comments and suggestions. The models in

342 this paper are provided courtesy of the AIM @SHAPE Shape  
 343 Repository. This work is partially supported by K.C. Wong  
 344 Magna Fund in Ningbo University, NSF of China (61571247,  
 345 61373068, U1636111), NSF of ZheJiang (LZ16F030001), NS-  
 346 F of Ningbo (2016A610041), the Opening Foundation of Zhe-  
 347 jiang Provincial Top Key Discipline(xkx11517), MOE2013-T2-  
 348 2-011 and MOE RG23/15.

## 349 References

350 [1] Ying X, Xin SQ, Sun Q, He Y. An intrinsic algorithm for parallel poisson  
 351 disk sampling on arbitrary surfaces. *IEEE Transactions on Visualization  
 352 & Computer Graphics* 2013;19(9):1425–37.

353 [2] Huang Q, Bart A, Martin W, Guibas LJ. Non-rigid registration under  
 354 isometric deformations. *Computer Graphics Forum* 2008;27(5):1449–57.

355 [3] Jin M. Computing geodesic spectra of surfaces. In: *ACM Symposium on  
 356 Solid and Physical Modeling*. 2007, p. 387–93.

357 [4] Liu YJ, Chen Z, Tang K. Construction of iso-contours, bisectors, and  
 358 Voronoi diagrams on triangulated surfaces. *IEEE Transactions on Pattern  
 359 Analysis & Machine Intelligence* 2010;33(8):1502–17.

360 [5] Xu C, Liu Y, Sun Q, Li J, He Y. Polyline-sourced geodesic Voronoi  
 361 diagrams on triangle meshes. *Comput Graph Forum* 2014;33(7):161–70.

362 [6] Liu Y, Xu C, Yi R, Fan D, He Y. Manifold differential evolution (MDE): a  
 363 global optimization method for geodesic centroidal Voronoi tessellations  
 364 on meshes. *ACM Trans Graph* 2016;35(6):243:1–243:10.

365 [7] Liu Y, Xu C, Fan D, He Y. Efficient construction and simplification of  
 366 Delaunay meshes. *ACM Trans Graph* 2015;34(6):174:1–174:13.

367 [8] Troy F, Attali D. From a closed piecewise geodesic to a constriction on a  
 368 closed triangulated surface. In: *Pacific Conference on Computer Graphics  
 369 and Applications*. 2003, p. 394.

370 [9] Peyré G, Cohen L. Surface segmentation using geodesic centroidal tesse-  
 371 lation. In: *International Symposium on 3D Data Processing, Visualization  
 372 and Transmission*, 2004. 3dpvt 2004. Proceedings. 2004, p. 995–1002.

373 [10] Sun Q, Zhang L, Zhang M, Ying X, Xin S, Xia J, et al. Texture brush: an  
 374 interactive surface texturing interface. In: *Proceedings of ACM Symposi-  
 375 um on Interactive 3D Graphics and Games, I3D '13*. 2013, p. 153–60.

376 [11] Sander PV, Wood ZJ, Gortler SJ, Snyder J, Hoppe H. Multi-chart geom-  
 377 etry images. In: *Eurographics/ACM SIGGRAPH Symposium on Geom-  
 378 etry Processing*. 2003, p. 146–55.

379 [12] Zhou K, Synder J, Guo B, Shum HY. Iso-charts: stretch-driven mesh  
 380 parameterization using spectral analysis. In: *Eurographics/ACM SIG-  
 381 GRAPH Symposium on Geometry Processing*. 2004, p. 45–54.

382 [13] Liu Y, Yu C, Yu M, He Y. Manifold SLIC: A fast method to compute  
 383 content-sensitive superpixels. In: *Proceedings of IEEE CVPR'16*. 2016,  
 384 p. 651–9.

385 [14] Mitchell JSB. *Geometric shortest paths and network optimization hand-  
 386 book of computational geometry*. Elsevier Science Publishers B V  
 387 2015;:49–119.

388 [15] Kimmel R, Sethian JA. Computing geodesic paths on manifolds. *Pro-  
 389 ceedings of the National Academy of Sciences of the United States of  
 390 America* 1998;95(15):8431–5.

391 [16] Mitchell JSB, Mount DM, Papadimitriou CH. The discrete geodesic prob-  
 392 lem. *SIAM J Comput* 1987;16(4):647–68.

393 [17] Chen J, Han Y. Shortest paths on a polyhedron. In: *Proceedings of  
 394 the sixth annual symposium on Computational geometry*. ACM; 1990, p.  
 395 360–9.

396 [18] Surazhsky V, Surazhsky T, Kirsanov D, Gortler SJ, Hoppe H. Fast exact  
 397 and approximate geodesics on meshes. *ACM Transactions on Graphics*  
 398 2005;24(3):553–60.

399 [19] Weber O, Devir YS, Bronstein AM, Bronstein MM, Kimmel R. Parallel  
 400 algorithms for approximation of distance maps on parametric surfaces.  
 401 *ACM Trans Graph* 2008;27(4).

402 [20] Xin SQ, Wang GJ. Improving Chen and Han's algorithm on the discrete  
 403 geodesic problem. *ACM Transactions on Graphics* 2009;28(4):104.

404 [21] Ying X, Xin SQ, He Y. Parallel Chen-Han (PCH) algorithm for discrete  
 405 geodesics. *ACM Transactions on Graphics* 2014;33(1):57–76.

406 [22] Xu C, Wang TY, Liu Y, Liu L, He Y. Fast wavefront propagation (FW-  
 407 P) for computing exact geodesic distances on meshes. *IEEE Trans Vis  
 408 Comput Graph* 2015;21(7):822–34.

409 [23] Qin Y, Han X, Yu H, Yu Y, Zhang J. Fast and exact discrete geodesic  
 410 computation based on triangle-oriented wavefront propagation. *ACM  
 411 Transactions on Graphics* 2016;35(4).

412 [24] Martínez D, Velho L, Carvalho PC. Computing geodesics on triangular  
 413 meshes. *Computers & Graphics* 2005;29(5):667–75.

414 [25] Kanai T, Suzuki H. Approximate shortest path on a polyhedral surface  
 415 based on selective refinement of the discrete graph and its applications.  
 416 In: *Geometric Modeling and Processing*. 2000, p. 241–50.

417 [26] Xin SQ, Wang GJ. Efficiently determining a locally exact shortest path  
 418 on polyhedral surfaces. *Computer-Aided Design* 2007;39(12):1081–90.

419 [27] Liu DC, Nocedal J. On the limited memory BFGS method for large scale  
 420 optimization. *Mathematical Programming* 1989;45(3):503–28.

421 [28] Sharir M, Schorr A. On shortest paths in polyhedral spaces. *SIAM J  
 422 Comput* 1986;15(1):193–215.

423 [29] Peyré G, Cohen LD. Heuristically driven front propagation for  
 424 fast geodesic extraction. *Lecture Notes in Computer Science*  
 425 2008;3752(1):173–85.

426 [30] Crane K, Weischedel C, Wardetzky M. Geodesics in heat: A new ap-  
 427 proach to computing distance based on heat flow. *ACM Transactions on  
 428 Graphics* 2013;32(5):152.

429 [31] Campen M, Heistermann M, Kobbelt L. Practical anisotropic geodesy.  
 430 *Comput Graph Forum* 2013;32(5):63–71.

431 [32] Solomon J, Rustamov R, Guibas L, Butscher A. Earth mover's distances  
 432 on discrete surfaces. *ACM Trans Graph* 2014;33(4):67:1–67:12.

433 [33] Lipman Y, Rustamov RM, Funkhouser TA. Biharmonic distance. *ACM  
 434 Transactions on Graphics* 2010;29(3):483–96.

435 [34] Aleksandrov L, Lanthier M, Maheshwari A, Sack JR. An  $\varepsilon$ -  
 436 approximation for weighted shortest paths on polyhedral surfaces. In:  
 437 *Proceedings of the 6th Scandinavian Workshop on Algorithm Theory  
 438 (SWAT '98)*. 1998, p. 11–22.

439 [35] Xin SQ, Ying X, He Y. Constant-time all-pairs geodesic distance query  
 440 on triangle meshes. In: *Proceedings of the ACM SIGGRAPH Symposium  
 441 on Interactive 3D Graphics and Games*. 2012, p. 31–8.

442 [36] Ying X, Wang X, He Y. Saddle vertex graph (SVG): A novel solu-  
 443 tion to the discrete geodesic problem. *ACM Transactions on Graphics*  
 444 2013;32(6):170:1–12.

445 [37] Wang X, Fang Z, Wu J, Xin SQ, He Y. Discrete geodesic graph (DGG) for  
 446 computing geodesic distances on polyhedral surfaces. *Computer Aided  
 447 Geometric Design* 2017;52:262–84.

448 [38] Xin SQ, He Y, Fu CW. Efficiently computing exact geodesic loops within  
 449 finite steps. *IEEE Transactions on Visualization and Computer Graphics*  
 450 2012;18(6):879–89.

451 [39] Wu C, Tai X. A level set formulation of geodesic curvature flow on simpli-  
 452 cial surfaces. *IEEE Transactions on Visualization and Computer Graphics*  
 453 2010;16(4):647–62.

454 [40] Mukhambetzhonov SG, Romashov YP, Sidorin SG, Tsentovskii EM.  
 455 Geodesic winding on conical surfaces of arbitrary profile. *Mechanics  
 456 of Composite Materials* 1993;28(6):540–5.

457 [41] Xin SQ, Chen S, Zhao J, Pan Z. Measuring length and girth of a tubular  
 458 shape by quasi-helices. *Computers & Graphics* 2014;38(1):392–8.

459 [42] Xin SQ, Wang W, Chen S, Zhao J, Shu Z. Intrinsic girth function for  
 460 shape processing. *ACM Transactions on Graphics* 2015;35(3):1–14.