Approximate Shortest Paths in Anisotropic Regions

Siu-Wing Cheng [*] Hyeon-Suk Na[†] Antoine Vigneron[‡] Yajun Wang[*]

October 9, 2007

Abstract

Our goal is to find an approximate shortest path for a point robot moving in a planar subdivision with n vertices. Let $\rho \geqslant 1$ be a real number. Distances in each face of this subdivision are measured by a convex distance function whose unit disk is contained in a concentric unit Euclidean disk, and contains a concentric Euclidean disk with radius $1/\rho$. Different convex distance functions may be used for different faces, and obstacles are allowed. These convex distance functions may be asymmetric. For any $\varepsilon \in (0,1)$ and for any two points v_s and v_d, we give an algorithm that finds a path from v_s to v_d whose cost is at most $(1+\varepsilon)$ times the optimal. Our algorithm runs in $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$ time. This bound does not depend on any other parameters; in particular it does not depend on the minimum angle in the subdivision. We give applications to two special cases that have been considered before: the weighted region problem and motion planning in the presence of uniform flows. For the weighted region problem with weights in $[1,\rho] \cup \{\infty\}$, the time bound of our algorithm improves to $O\left(\frac{\rho \log \rho}{\varepsilon} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$.

1 Introduction

The problem of computing a shortest path between two points arises naturally in geographic information systems, VLSI design, logistics, and motion planning. Another area of application for shortest paths algorithms is computer graphics, where the geometric properties of shortest paths along a surface can be exploited for mesh cutting and editing. (See the article by Surazhsky et al. [20].) The path lies in a geometric environment in all these applications. This environment is usually represented by a polygonal (or polyhedral) subdivision. Different metrics may be used in different regions of the subdivision in order to model friction, wind, steepness, or any other mechanical constraint.

Due to these applications, and the variety of possible geometric environments and metrics, algorithms for geometric shortest paths problems have been extensively studied. We mention the most relevant work here and refer the interested reader to the survey by Mitchell [12] for more details.

[*]Department of Computer Science and Engineering, HKUST, Hong Kong. Email: {scheng,yalding}@cse.ust.hk

[†]Research supported by KRF funded by the Korean Government (R04-2004-000-10004-0). School of Computing, Soongsil University, Seoul, Korea. Email: hsnaa@computing.ssu.ac.kr

[‡]Applied Mathematics and Informatics Department, INRA, Jouy-en-Josas, France. Email: antoine.vigneron@jouy.inra.fr

In the weighted region problem [13], a point robot moves within a planar subdivision $\mathcal{T}$, each face f of $\mathcal{T}$ being associated with a weight $w_f > 0$. The cost of a path within a face f is the length of this path multiplied by w_f. Assume that the faces of the subdivision $\mathcal{T}$ are all triangular. (We can make this assumption as any planar subdivision can be triangulated by introducing a linear number of edges.) We denote by n the number of vertices of $\mathcal{T}$. The first approximation scheme for the weighted region problem was given by Mitchell and Papadimitriou [13]. It runs in $O(n^8 L)$ time, where L represents the maximum number of bits of the input numbers (such as the integer coordinates of the vertices of the subdivision, and the weights). This algorithm is a continuous version of the Dijkstra's algorithm for finding shortest paths in a graph. Other algorithms for the weighted region problem discretize the search space by placing Steiner points, and by finding a shortest path in a graph whose nodes are Steiner points or input vertices, and whose edges are line segments. In particular, Aleksandrov et al. [2] and Sun and Reif [19] gave algorithms that have linear dependency in n. However, their time bounds depend on the minimum angle in $\mathcal{T}$ (and the weights too in the algorithm by Aleksandrov et al. [2]).

The main limitation of the weighted region model is that it only models situations where the metrics are isotropic. It cannot account for the effect of wind, current, or any other force field that favors some directions of travel. A more general model was introduced by Reif and Sun for motion planning in the presence of uniform flows [16]. In each face f of $\mathcal{T}$, the velocity of the robot is the sum of a flow $\vec{v}_f$ and a control velocity chosen by the robot. It allows one to model friction and a uniform flow within each region. Reif and Sun [16] showed that this problem is PSPACE hard in 3D, and gave a FPTAS for the 2D case. (Some geometric parameters were treated as constant, such as the minimum angle in $\mathcal{T}$).

As pointed out by Aleksandrov et al. [2], one challenge is to remove the dependence on parameters other than n and ε in the running time. It is also desirable to handle more general types of metrics in order to model a larger class of problems that arise in applications. (See the article of Sellen [17] on the direction-weighted problem, where the cost is proportional to a continuous function of the direction.) We make progress in both aspects in this paper.

1.1 Our Results

We consider a generalization of Reif and Sun's model for motion planning in the presence of uniform flows [16]. A point robot moves within a planar subdivision from a source point v_s to a target point v_d. The planar subdivision $\mathcal{T}$ may have holes in order to model obstacles. The distance within each face f of the subdivision is measured according to a possibly asymmetric convex distance function [4]. (See Section 2.2 for a definition of convex distance functions.) Different convex distance functions may be used for different faces. The cost of a path is measured according to these distance functions. (See Section 2 for the case of a polygonal path and Section 6 for the case of a rectifiable path.) Let B_f denote the unit "disk" of the distance function of a face f. We assume that B_f is contained in a concentric unit Euclidean disk and that B_f contains a concentric Euclidean disk with radius $1/\rho$. In other words, there exists $\rho \geqslant 1$ such that the speed of the robot in any direction and within any face of $\mathcal{T}$ is in the interval $[1/\rho, 1]$. The weighted region problem and the problem of path planning in the presence of uniform flows are special cases in our model.

We assume that the distance between two points under any of the convex distance functions can be computed in $O(1)$ time. Our model of computation is the standard real–RAM model [15] in which the operations $(+, -, \times, /)$ can be performed in constant time.

Our main results include an algorithm that computes in time $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$ a polygonal path whose cost is at most $(1+\varepsilon)$ times the optimal. This is the first algorithm that can handle general convex distance functions. For the weighted region problem, the time bound improves to $O\left(\frac{\rho \log \rho}{\varepsilon} n^3 \log\left(\frac{n\rho}{\varepsilon}\right)\right)$. Our time bounds have the nice feature that they do not depend on the geometry of $\mathcal{T}$. It is not obvious that an optimal path exists in our model. Indeed, we give an example in which no polygonal path is optimal. Nevertheless, using the theory of length spaces [3, 10], we can prove that there exists an optimal *rectifiable* path. (A rectifiable path is a path with finite Euclidean length.) Furthermore, we show that there exists a $(1+\varepsilon)$-approximate shortest path that is polygonal and has $O(\rho n^2/\varepsilon)$ links. This is instrumental to establishing the correctness and the complexity of our algorithm.

Our approach is the following. We define a k-link path to be a polygonal path with at most k edges, each edge being contained in a face of $\mathcal{T}$ and having its endpoints on the boundary of this face. (Our definition is different from the one given by Daescu et al. [5].) We first give approximation algorithms that return a polygonal path with cost at most $(1+\varepsilon)$ times the cost of any k-link path. (See Section 3.) In the weighted region problem, Mitchell and Papadimitriou [13] showed that there exists a shortest path that is a k-link path with $k = \Theta(n^2)$, so we apply our algorithm with $k = \Theta(n^2)$ and obtain the result stated above. In the general case, we show that for any polygonal path P, there exists a $(21\rho n^2/\varepsilon)$-link path with cost at most $(1+\varepsilon)$ times the cost of P. (See Sections 4 and 5.) Thus, by choosing $k = 21\rho n^2/\varepsilon$, our algorithm returns a path with cost at most $(1+\varepsilon)$ times the cost of any polygonal path. Then, in Section 6, we prove that there exists an optimal rectifiable path, and show that there exists a polygonal path with cost arbitrarily close to the optimal. It follows that our algorithm returns a $(1+\varepsilon)$ approximate shortest path within the class of rectifiable paths.

1.2 Comparison with Previous Work

A direct comparison cannot be made with any previous result because our algorithm is the first that can handle general convex distance functions. Anisotropic shortest paths problems have been studied on terrains [11, 18] in a special case that models some common mechanical constraints on a mobile robot. This model cannot account for the presence of flows (even uniform flows), so it is not more general than ours. On the other hand, our algorithm applies to planar subdivisions, not terrains. The problems of navigating through weighted regions and in the presence of uniform flows are special cases in our model. So we compare with the previous results for these problems.

Weighted Region Problem. In this case, ρ is equal to the ratio of the maximum weight to the minimum weight. Other than the dependence on n and ε, there is a factor of $\Omega(1/\theta_{\min})$ in the worst-case running times of the algorithms by Aleksandrov et al. [1, 2] and Sun and Reif [19], where $\theta_{\min}$ is the minimum angle in $\mathcal{T}$. The worst-case running time of the algorithms by Aleksandrov et al. [1, 2] depend on ρ too. Our running time is independent of the geometry of $\mathcal{T}$, although it has a higher dependence on n, ε, and ρ. The algorithm of Mitchell and Papadimitriou [13] has a running time of $O(n^8 \log \frac{nN\rho}{\varepsilon})$, where the input vertices have integer coordinates in $[0, N]$. In comparison, our algorithm works in the standard real RAM model; and our algorithm has a smaller dependence on n, but a higher dependence on ε and ρ.

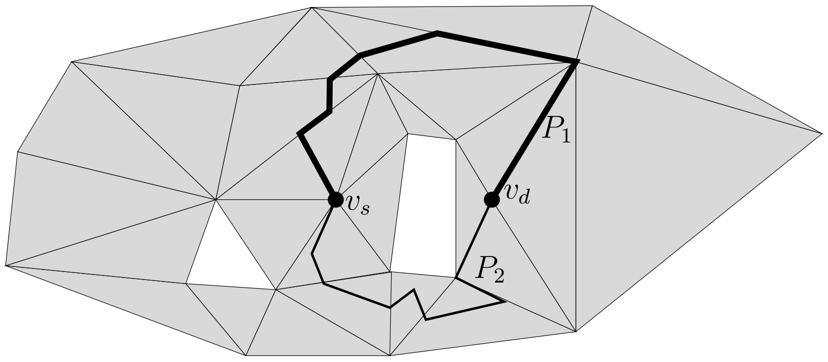

Figure 1: The underlying space $|\mathcal{T}|$ is shaded. The path P_1 is a 7-link path, and thus it is $\mathcal{T}$-respecting. The path P_2 is a $\mathcal{T}$-respecting path with 8 links, but it is not an 8-link path.

Movement in the Presence of Uniform Flows. Assume that there is a uniform flow of velocity $\vec{v}_f$ in each face f of $\mathcal{T}$. The robot can apply a control velocity $\vec{v}_r$ in any direction such that $\|\vec{v}_r\|$ is at most some constant c_f. For each face f, we define a convex distance function whose unit "disk" B_f is a Euclidean disk with radius c_f and centered at $O + \vec{v}_f$, where O is the origin. Thus, B_f is contained in a disk with radius $c_f + \|\vec{v}_f\|$ and centered at O, and B_f contains a disk with radius $c_f - \|\vec{v}_f\|$ and centered at O. Let $v_{\min} = \min\{c_f - \|\vec{v}_f\| : f \in \mathcal{T}\}$. Let $v_{\max} = \max\{c_f + \|\vec{v}_f\| : f \in \mathcal{T}\}$. Assuming that $v_{\min} > 0$, we can model the problem with $\rho = v_{\max}/v_{\min}$, and find an approximate shortest path in time $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$. An algorithm by Reif and Sun runs in $O\left(\frac{nC_{\text{skew}}}{\varepsilon}(\log \frac{C_{\text{skew}}}{\varepsilon})(\log \frac{C_{\text{skew}}}{\varepsilon} + \log n)\right)$ time [16], where C_{skew} is defined as follows. Let $\lambda = \max\{c_f/c_{f'} : \text{adjacent faces } f \text{ and } f'\}$. Let $\rho_{\min} = \min\{c_f/\|\vec{v}_f\| : f \in \mathcal{T}\}$. Let $\theta_{\min}$ be the minimum angle in $\mathcal{T}$. Then $C_{\text{skew}} = \Theta\left(\frac{\lambda(\rho_{\min}+1)}{\theta_{\min}(\rho_{\min}-1)}\right)$. Reif and Sun's algorithm requires $\rho_{\min} > 1$, which is equivalent to our condition of $v_{\min} > 0$. Our running time does not depend on $\theta_{\min}$, but Reif and Sun's running time has a better dependence on n, ε, and ρ.

2 Notation and Preliminaries

2.1 Environment

We model the environment $\mathcal{T}$ in which the point robot can move by a *simplicial complex* [6]: it is a collection of triangles such that any two triangles can only intersect along a common edge or vertex or not at all. A *face* of $\mathcal{T}$ is a triangle in $\mathcal{T}$. We consider faces to be closed subsets of $\mathbb{R}^2$. A *vertex* (resp. an *edge*) of $\mathcal{T}$ is a vertex (resp. an edge) of some face of $\mathcal{T}$. (We do not allow dangling edges or vertices; that is, all edges and vertices must be edges or vertices of a face of $\mathcal{T}$.) The *underlying space* $|\mathcal{T}| \subset \mathbb{R}^2$ of $\mathcal{T}$ is the union of its faces. (See Figure 1). We assume that $|\mathcal{T}|$ is connected, but we allow $|\mathcal{T}|$ to have holes. The point robot is free to move inside $|\mathcal{T}|$ but it cannot move outside $|\mathcal{T}|$. We use n to denote the number of vertices in $\mathcal{T}$, and we assume that n is finite.

For any two points $p, q \in \mathbb{R}^2$, we denote by $\overline{pq}$ the closed, oriented line segment from p to q. In particular, when $p \neq q$, we have $\overline{pq} \neq \overline{qp}$. We denote by $\|pq\|$ the Euclidean distance between p and q. For any two points $p, q \in |\mathcal{T}|$, the *geodesic distance* between p and q is the Euclidean length of the shortest polyline in $|\mathcal{T}|$ with endpoints p and q. We use $\|pq\|_{\mathcal{T}}$ to denote this geodesic distance.

For each face f of $\mathcal{T}$, we denote by $\mathrm{int}(f)$ (resp. $\mathrm{bd}(f)$) the interior (resp. boundary) of f according to the usual topology of $\mathbb{R}^2$. For a segment $\overline{pq}$, we use $\mathrm{int}(\overline{pq})$ to denote the open line segment from p to q. When X is a polyline with endpoints p, q, we use $\mathrm{int}(X)$ to denote $X \setminus \{p, q\}$. A *chord* of a face f of $\mathcal{T}$ is an oriented line segment $\overline{pq}$ such that $\mathrm{int}(\overline{pq}) \subset \mathrm{int}(f)$ and $\{p, q\} \subset \mathrm{bd}(f)$. If $\overline{pq} \subset \mathrm{bd}(f)$, we say that $\overline{pq}$ is a *boundary segment*.

2.2 Convex Distance Functions

Each face f of $\mathcal{T}$ is associated with a compact convex set $B_f \subset \mathbb{R}^2$ that contains the origin O in its interior. The *convex distance function* associated with f is defined by

$$\forall\, x, y \in f,\ d_f(x, y) = \min\{\lambda \in [0, +\infty) : y \in x + \lambda B_f\}.$$

This type of distance function has been studied before [4, 7] in the context of Voronoi diagrams. A convex distance function is not necessarily a metric as it may be asymmetric. Still, d_f satisfies the triangle inequality, and the shortest path from p to q is the oriented line segment $\overline{pq}$.

If $\mathrm{int}(\overline{pq}) \subset \mathrm{int}(f)$ for some face f, the cost of $\overline{pq}$ is defined as $\mathrm{cost}(\overline{pq}) = d_f(p, q)$. If $\overline{pq}$ is contained in an edge e of $\mathcal{T}$ that is adjacent to exactly one face f, we also define $\mathrm{cost}(\overline{pq})$ to be $d_f(p, q)$. On the other hand, if e is adjacent to two faces f_1 and f_2, we define $\mathrm{cost}(\overline{pq})$ to be $\min(d_{f_1}(p, q), d_{f_2}(p, q))$.

We assume that there exists $\rho \geqslant 1$ such that, for any face f, the set B_f contains a Euclidean disk with radius $1/\rho$ centered at the origin, and B_f is contained in the unit Euclidean disk centered at the origin. Intuitively, it means that the speed allowed in any direction is always in the interval $[1/\rho, 1]$. It implies that, for any face f and for any points $p, q \in f$, we have $\|pq\| \leqslant \mathrm{cost}(\overline{pq}) \leqslant \rho\,\|pq\|$. Another useful consequence is that for any two chords $\overline{ps}$ and $\overline{qr}$ of the same face f, we have

$$\begin{aligned} \mathrm{cost}(\overline{ps}) = d_f(p, s) &\leqslant d_f(p, q) + d_f(q, r) + d_f(r, s) \\ &= d_f(p, q) + \mathrm{cost}(\overline{qr}) + d_f(r, s) \\ &\leqslant \rho\,\|pq\| + \mathrm{cost}(\overline{qr}) + \rho\,\|rs\|. \qquad (1) \end{aligned}$$

A similar derivation shows that the above inequality also holds when $\overline{ps}$ and $\overline{qr}$ are boundary segments contained in the same edge of $\mathcal{T}$. (If the edge containing $\overline{ps}$ and $\overline{qr}$ is incident to two faces f_1 and f_2, one needs to replace $d_f(x, y)$ by $\min(d_{f_1}(x, y), d_{f_2}(x, y))$ for any x and y in the above derivation.)

2.3 Polygonal Paths

We consider paths from a point v_s to a point v_d in $|\mathcal{T}|$. Without loss of generality, we can assume that v_s and v_d are vertices of $\mathcal{T}$. If not, we can force v_s and v_d to be vertices by splitting the triangle(s) containing them into smaller triangles. (These smaller triangles inherit their convex distance functions from the triangles containing them.)

A *polygonal path* is a polyline in $|\mathcal{T}|$ with endpoints v_s and v_d. A *link* is an edge of a polygonal path and a *node* is a vertex of a polygonal path—we use this terminology to avoid confusion with edges and vertices of $\mathcal{T}$. We identify a polygonal path with its sequence of nodes. So if a polygonal path P has the node sequence $(v_s = p_0, p_1, \ldots, p_m = v_d)$, we write $P = (p_0, p_1, \ldots, p_m)$. We do not require the nodes of P to be distinct. The *length* of P is defined as $\text{length}(P) = \sum_{i=1}^{m} \|p_{i-1}p_i\|$.

A $\mathcal{T}$*-respecting path* is a polygonal path P such that each link of P is contained in a face of $\mathcal{T}$. (See Figure 1). The cost of a $\mathcal{T}$-respecting path P is simply the sum of the costs of its links when P is traversed from v_s to v_d. Therefore, we have $\text{cost}(P) = \sum_{i=1}^{m} \text{cost}(\overline{p_{i-1}p_i})$. It implies that $\text{cost}(P)/\rho \leqslant \text{length}(P) \leqslant \text{cost}(P)$. We also define $\text{cost}(P)$ when P is an arbitrary polygonal path. We first obtain a $\mathcal{T}$-respecting path P' by introducing new nodes at the intersections between the links of P and the edges of $\mathcal{T}$. We then define $\text{cost}(P) = \text{cost}(P')$.

For any integer k, a k*-link path* is a polygonal path with at most k links, and whose links are either chords or boundary segments. (See Figure 1.) In particular, a k-link path is $\mathcal{T}$-respecting, and none of its nodes lies in the interior of a face. Our definition is different from previous work [5] on k-link paths, where the path is not required to be $\mathcal{T}$-respecting. By our definitions, a $\mathcal{T}$-respecting path can have a node in the interior of a face of $\mathcal{T}$, but a k-link path cannot.

3 Approximation Algorithms

We present algorithms for approximating a shortest k-link path for some given $\varepsilon \in (0, 1)$ and k. The cost of the output polygonal path is less than $(1 + \varepsilon)$ times the cost of any k-link path, but the output path is allowed to have more than k links. After showing the existence of an approximate shortest path with few links, we can use these algorithms to find $(1 + \varepsilon)$-approximate shortest paths.

We first present a simple algorithm in Section 3.1, which discretizes the environment with a graph and then computes a shortest path in the graph. Unlike the previous discretization schemes [1, 2], our discretization makes use of the global geodesic distance $\|v_s v_d\|_{\mathcal{T}}$. In Section 3.2, we show another simple idea to space out the graph vertices that are far away from v_s and v_d. It results in a reduction of the graph size and hence an improvement of the running time. We show that it is possible to improve the running time further by working with the graph vertices alone (without computing the graph edges), by employing the algorithm BUSHWHACK [19].

3.1 A Simple Algorithm

In this section we present a simple algorithm that computes an approximate k-link shortest path. It is based on two ideas. First, we observe that any $\frac{4}{3}$-approximate shortest path lies in a region delimited by an ellipse with diameter $\frac{4}{3}\rho \|v_s v_d\|_{\mathcal{T}}$. Second, we discretize the problem by placing Steiner points uniformly along the portion of each edge of $\mathcal{T}$ that lies inside this ellipse. With an appropriate spacing of the Steiner points, we show that there is an approximate shortest path whose nodes are all Steiner points or vertices of $\mathcal{T}$. We find one such path by computing a shortest path in a graph whose nodes are the Steiner points and the vertices of $\mathcal{T}$, using Dijkstra's algorithm.

Let $k \geqslant 2n - 4$ denote an integer. Because $\mathcal{T}$ is a planar graph with n vertices, it has at most $2n - 4$ faces [14]. It implies that there exists a k-link path from v_s to v_d. Therefore, there

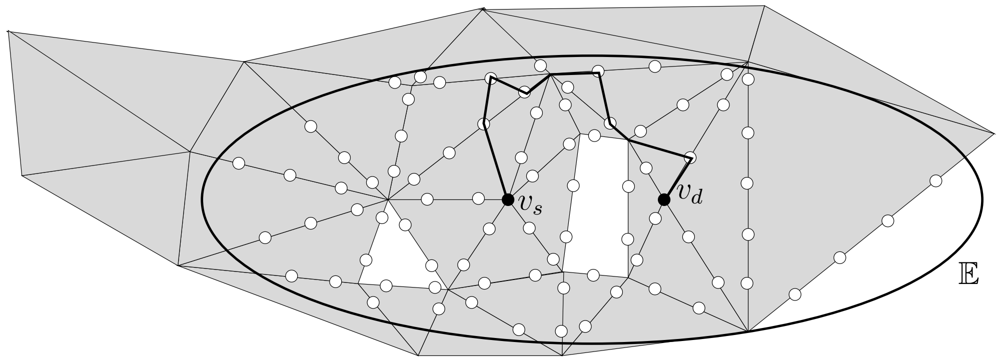

Figure 2: The white points are the Steiner points. The polygonal path is a 9-link path whose nodes are all Steiner points or vertices of $\mathcal{T}$.

exists a k-link path P_k^ε such that for any $\varepsilon > 0$,

$$\text{cost}(P_k^\varepsilon) \leqslant \left(1 + \frac{\varepsilon}{3}\right) \inf\{\text{cost}(P_k) : P_k \text{ is a } k\text{-link path}\}.$$

This path has the following property.

Lemma 3.1 *If $k \geqslant 2n - 4$ and $\varepsilon \in (0, 1)$, then* $\text{cost}(P_k^\varepsilon) \leqslant \dfrac{4\rho}{3} \|v_s v_d\|_{\mathcal{T}}$.

Proof. Let $G = (g_0, g_1, \ldots, g_m)$ be a $\mathcal{T}$-respecting path with length $\|v_s v_d\|_{\mathcal{T}}$ and such that m is minimum. The path G cannot have two links inside the same face of $\mathcal{T}$: if there were two such links, say $\overline{g_{i-1}g_i}$ and $\overline{g_j g_{j+1}}$ with $i \leqslant j$, we could remove from G the nodes $(g_i, \ldots, g_j)$. It would yield a polygonal path with at most the same length, but fewer nodes. As we noted above, $\mathcal{T}$ has at most $2n - 4$ faces. Therefore, G is a $(2n - 4)$-link path and hence a k-link path. We conclude that $\text{cost}(P_k^\varepsilon) \leqslant \left(1 + \frac{\varepsilon}{3}\right) \text{cost}(G) \leqslant \frac{4}{3} \text{cost}(G) = \frac{4}{3} \sum_{i=1}^m \text{cost}(\overline{g_{i-1}g_i}) \leqslant \frac{4}{3} \sum_{i=1}^m \rho \|g_{i-1}g_i\| \leqslant \frac{4\rho}{3} \text{length}(G) = \frac{4\rho}{3} \|v_s v_d\|_{\mathcal{T}}$. ◻

Let $\mathbb{E}$ denote the following elliptic region:

$$\mathbb{E} = \left\{ x \in \mathbb{R}^2 : \|v_s x\| + \|v_d x\| \leqslant \frac{4\rho}{3} \|v_s v_d\|_{\mathcal{T}} \right\}.$$

Since $\text{length}(P_k^\varepsilon) \leqslant \text{cost}(P_k^\varepsilon)$, we know by Lemma 3.1 that $P_k^\varepsilon \subset \mathbb{E}$. For each edge e of $\mathcal{T}$, we place a maximal set of equally spaced points on $\text{int}(e \cap \mathbb{E})$. (See Figure 2.) The spacing is $\delta = \frac{\varepsilon}{6\rho k} \|v_s v_d\|_{\mathcal{T}}$. The following lemma shows that our Steiner points give an accurate discretization.

Lemma 3.2 *If $k \geqslant 2n - 4$ and $\varepsilon \in (0, 1)$, there exists a k-link path S_k such that* $\text{cost}(S_k) \leqslant (1 + \varepsilon) \text{cost}(P_k)$ *for any k-link path P_k, and all the nodes of S_k are Steiner points or vertices of $\mathcal{T}$.*

Proof. Let $(p_0, p_1, \ldots, p_m)$ be the node sequence of P_k^ε. For any $i \in [0, m]$, we associate a point s_i to p_i as follows. If p_i is a vertex of $\mathcal{T}$, we set $s_i = p_i$. Otherwise, since P_k^ε is a k-link path,

p_i lies in the interior of some edge e of $\mathcal{T}$. Thus, there is a Steiner point $x \in \text{int}(e)$ such that $\|xp_i\| \leqslant \delta$. We set $s_i = x$. We denote $S_k = (s_0, s_1, \ldots, s_m)$. (It is possible that $s_{i-1} = s_i$ for some i, in which case the link $\overline{s_{i-1}s_i}$ degenerates to a point.) By construction, for any $i \in [1, m]$, the links $\overline{p_{i-1}p_i}$ and $\overline{s_{i-1}s_i}$ either are chords of the same face of $\mathcal{T}$, or are contained in the same edge of $\mathcal{T}$. Therefore, S_k is a k-link path, and inequality (1) implies that

$$\begin{aligned}\text{cost}(S_k) &\leqslant \sum_{i=1}^{m} \rho\,\|s_{i-1}p_{i-1}\| + \text{cost}(\overline{p_{i-1}p_i}) + \rho\,\|s_ip_i\| && (2)\\ &\leqslant \text{cost}(P_k^{\varepsilon}) + 2\rho m\delta && (3)\\ &= \text{cost}(P_k^{\varepsilon}) + \frac{m\varepsilon}{3k}\,\|v_sv_d\|_{\mathcal{T}} && (4)\end{aligned}$$

By definition, P_k^{ε} is a k-link path. So $m \leqslant k$, and thus $\text{cost}(S_k) \leqslant \text{cost}(P_k^{\varepsilon}) + \frac{\varepsilon}{3}\,\|v_sv_d\|_{\mathcal{T}}$. Since $\|v_sv_d\|_{\mathcal{T}} \leqslant \text{length}(P_k^{\varepsilon}) \leqslant \text{cost}(P_k^{\varepsilon})$, we have $\text{cost}(S_k) \leqslant (1+\varepsilon/3)\,\text{cost}(P_k^{\varepsilon})$. Thus, for any k-link path P_k, $\text{cost}(S_k) \leqslant (1+\varepsilon/3)^2\,\text{cost}(P_k) \leqslant (1+\varepsilon)\,\text{cost}(P_k)$. ◻

The *Steiner graph* is the directed weighted graph defined as follows. Its nodes are the Steiner points and the vertices of $\mathcal{T}$. There is a directed edge (p, q) between any two nodes p and q that lie on the boundary of the same face of $\mathcal{T}$. The edge (p, q) is assigned the weight $\text{cost}(\overline{pq})$.

Here is the pseudo-code of our approximation algorithm.

```
Approximate($\mathcal{T}, k, \varepsilon$)
    1. Compute $\|v_sv_d\|_{\mathcal{T}}$.
    2. Compute the Steiner graph.
    3. Compute a weighted shortest path $S$ in the Steiner graph.
    4. Output $S$.
```

In the analysis of this algorithm, we use the standard real-RAM model [15] and assume that for any points p and q in the same face of $\mathcal{T}$, we can compute $\text{cost}(\overline{pq})$ in $O(1)$ time. We obtain the following result:

Lemma 3.3 *If* $k \geqslant 2n-4$ *and* $\varepsilon \in (0,1)$*, then* Approximate($\mathcal{T}, k, \varepsilon$) *computes a polygonal path* S *such that* $\text{cost}(S) \leqslant (1+\varepsilon)\,\text{cost}(P_k)$ *for all* k*-link path* P_k*. The algorithm can be implemented to run in* $O\left(nk^2\rho^4/\varepsilon^2\right)$ *time.*

Proof. Correctness follows from Lemma 3.2. The geodesic distance computation in Line 1 can be performed in $O(n^2) = O(nk)$ time [12]. The diameter of $\mathbb{E}$ is $\frac{4}{3}\rho\,\|v_sv_d\|_{\mathcal{T}}$, so each edge e of $\mathcal{T}$ contains $O(k\rho^2/\varepsilon)$ Steiner points. Therefore, the Steiner graph has $O(nk\rho^2/\varepsilon)$ nodes and $O(nk^2\rho^4/\varepsilon^2)$ edges. Using a Fibonacci heap, Dijkstra's algorithm can be implemented to compute single-source shortest paths in $O(|E| + |V|\log|V|)$ time for a graph with $|V|$ nodes and $|E|$ edges [8]. We use this method to implement step 4 and, thus, we obtain the desired running time. ◻

3.2 A Sparser Steiner Graph

We can speed up the above algorithm by reducing the number of Steiner points. The idea is to place Steiner points more sparsely on portions of edges that are far from v_s and v_d. So we introduce a family of elliptic regions. For $0 \leqslant i \leqslant \lceil \log \rho \rceil$, each region is defined as

$$\mathbb{E}_i = \{\, x \in \mathbb{R}^2 : \|v_s x\| + \|v_d x\| \leqslant \frac{4\rho}{2^i\, 3} \|v_s v_d\|_{\mathcal{T}} \,\}.$$

For convenience, we take $\mathbb{E}_{\lceil \log \rho \rceil + 1}$ to denote the empty set.

We construct a set of Steiner points as follows. For each edge e of $\mathcal{T}$ and for any integer i such that $0 \leqslant i \leqslant \lceil \log \rho \rceil$, we insert the intersection points between e and the boundary of $\mathbb{E}_i$, and then we place a maximal set of points on $\operatorname{int}(e \cap (\mathbb{E}_i \setminus \mathbb{E}_{i+1}))$ with uniform spacing $\delta_i = \frac{\varepsilon}{2^{i+1}\, 6k} \|v_s v_d\|_{\mathcal{T}}$. After obtaining the new set of Steiner points, the Steiner graph is constructed as before. Observe that now, the number of Steiner graph nodes per edge is $O((k\rho \log \rho)/\varepsilon)$, instead of $O(k\rho^2/\varepsilon)$. Therefore, the size of the Steiner graph is $O((nk^2\rho^2 \log^2 \rho)/\varepsilon^2)$ and the running time improves to $O((nk^2\rho^2 \log^2 \rho)/\varepsilon^2)$.

It remains to prove that this new algorithm is correct. Let j denote the largest integer such that $P_k^\varepsilon \subset E_j$. (There is such an integer because $P_k^\varepsilon \subset \mathbb{E}_0 = \mathbb{E}$ by Lemma 3.1.) For every edge e of $\mathcal{T}$, the distance between two consecutive Steiner points along $e \cap E_j$ is at most δ_j. Thus, using the derivation in inequalities (2) and (3) in the proof of Lemma 3.2, we can show that $\operatorname{cost}(S_k) \leqslant \operatorname{cost}(P_k^\varepsilon) + \rho\varepsilon \|v_s v_d\|_{\mathcal{T}}/(2^{j+1}\, 3)$. As P_k^ε is not contained in $\mathbb{E}_{j+1}$, we have $\operatorname{length}(P_k^\varepsilon) \geqslant 4\rho \|v_s v_d\|_{\mathcal{T}}/(2^{j+1}\, 3)$ and thus $\operatorname{cost}(S_k) \leqslant (1 + \varepsilon/4) \operatorname{cost}(P_k^\varepsilon)$. Recall that $\operatorname{cost}(P_k^\varepsilon) \leqslant (1+\varepsilon/3) \operatorname{cost}(P_k)$ for any k-link path P_k. It follows that $\operatorname{cost}(S_k) \leqslant (1+\varepsilon) \operatorname{cost}(P_k)$. Therefore, we have proved the following result:

Lemma 3.4 *If $k \geqslant 2n - 4$ and $\varepsilon \in (0, 1)$, we can compute in $O((nk^2\rho^2 \log^2 \rho)/\varepsilon^2)$ time a polygonal path S such that* $\operatorname{cost}(S) \leqslant (1 + \varepsilon) \operatorname{cost}(P_k)$ *for any k-link path P_k.*

3.3 Further Improvement

We can improve the running time further if we can avoid computing explicitly the edges of the Steiner graph. It means that, instead of using Dijkstra's algorithm, we need to use an algorithm that does not require these edges to produce a shortest path. BUSHWHACK, an algorithm by Sun and Reif [19], does exactly this, provided that the cost function is *pseudo-Euclidean.*

The cost function is pseudo-Euclidean if it meets the following two conditions. First, it obeys the triangle inequality in the interior of each face (in other words, a shortest path in the interior of a face is a line segment). Second, let p be a point inside a face f and let e be an edge of f such that $p \notin e$. Let $\varphi_{p,e} : e \to \mathbb{R}$ be the function defined by $\varphi_{p,e}(x) = \operatorname{cost}(\overline{px})$. For any p and e, this function is required to have the following property: we can compute in $O(1)$ time a partition of e into $O(1)$ subsegments such that $\varphi_{p,e}$ is monotone along each such subsegment.

The convexity of the distance functions in $\mathcal{T}$ implies that $\varphi_{p,e}$ has only one local extremum, which is a global minimum. (In degenerate cases, when B_f is not strictly convex, this minimum can be achieved over a closed segment and not just at a single point). So if we assume that this minimum can be computed in $O(1)$ time, our metric is pseudo-Euclidean.

When there are m Steiner points per edge, the BUSHWHACK algorithm runs in time $O(mn \log(mn))$, instead of $O(m^2 n + mn \log(mn))$ for Dijkstra's algorithm. The algorithm presented in Section 3.2 uses $m = O((k\rho \log \rho)/\varepsilon)$ Steiner points per edge. So, we obtain this improved algorithm:

Theorem 1 *If $k \geqslant 2n - 4$ and $\varepsilon \in (0,1)$, we can compute in time $O\left(\frac{nk\rho \log \rho}{\varepsilon} \log\left(\frac{k\rho}{\varepsilon}\right)\right)$ a polygonal path S such that $\operatorname{cost}(S) \leqslant (1+\varepsilon)\operatorname{cost}(P_k)$ for any k-link path P_k.*

3.4 Applications

In the weighted region problem, each face f is associated with a weight $w_f \in [1, \rho]$ (assuming that the minimum weight is scaled to 1). According to our terminology, for each face f, the set B_f is the Euclidean disk centered at the origin with radius $1/w_f$. Obstacles (holes in $|\mathcal{T}|$) are still allowed; in other words, we allow weights to be in $[1, \rho] \cup \{+\infty\}$. Mitchell and Papadimitriou [12, 13] proved that, in the weighted region problem, the shortest path is an $O(n^2)$-link path. Substituting this bound for k into Theorem 1 yields the following result:

Corollary 1 *Consider the weighted region problem in a planar subdivision with n vertices and with weights in $[1,\rho] \cup \{+\infty\}$. For any $\varepsilon \in (0,1)$, we can compute a $(1+\varepsilon)$-approximate shortest path in time $O\left(\frac{\rho \log \rho}{\varepsilon} n^3 \log\left(\frac{n\rho}{\varepsilon}\right)\right)$.*

To realize the $1+\varepsilon$ approximation bound in Corollary 1, one would need to extract from [13] the constant c_0 hidden in the $O(n^2)$ bound on the number of links. Nevertheless, the proof of Lemma 3.2 reveals that even if k is set to be $c_1 n^2$ for some constant $c_1 < c_0$, the approximation ratio is still $1 + O(\varepsilon)$.

In the anisotropic setting where each face f is associated with a convex distance function d_f, no bound on the number of links in the shortest path was known before. In Theorem 2 in Section 5 and Corollary 4 in Section 6, for any $\varepsilon \in (0,1)$, we prove a bound of $21\rho n^2/\varepsilon$ on the number of links in a polygonal path whose cost is at most $(1+\varepsilon)$ times the optimal. Substituting this bound for k into Theorem 1 yields the following result:

Corollary 2 *Consider the anisotropic shortest path problem in a planar subdivision with n vertices. For any $\varepsilon \in (0,1)$, we can compute a $(1+\varepsilon)$-approximate shortest path in time $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$.*

As we mentioned in the introduction (Section 1.2), the shortest path problem in the presence of uniform flows can be solved using Corollary 2.

Corollary 3 *Consider the shortest path problem in the presence of uniform flows as defined in Section 1.2. Assume that $v_{\min} > 0$. For any $\varepsilon \in (0,1)$, we can compute a $(1+\varepsilon)$-approximate shortest path in time $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$, where $\rho = v_{\max}/v_{\min}$.*

4 Cost-Preserving Path Transformations

In this section we show how to transform a $\mathcal{T}$-respecting path into another $\mathcal{T}$-respecting path that has at most the same cost and is free of some undesirable features. This result is used in Section 5 to show that any polygonal path can be converted into a polygonal path with $O(\rho n^2/\varepsilon)$ links such that the cost increases by a factor of at most $1+\varepsilon$.

4.1 Preliminaries

Let $P = (v_s = p_0, \dots, p_m = v_d)$ be a polygonal path. We allow $p_i = p_{i+1}$; in this case, we call $\overline{p_i p_{i+1}} = \{p_i\}$ a *degenerate link*. The path P is *self-intersecting* if

- there exist $i < j$ such that $\overline{p_{i-1}p_i} \cap \overline{p_j p_{j+1}} \neq \emptyset$, or
- there exists i such that $p_{i-1} \in \overline{p_i p_{i+1}}$ or $p_{i+1} \in \overline{p_{i-1}p_i}$.

The path P is *simple* if it is not self-intersecting. The points in which P self-intersects are not necessarily nodes of P.

To describe the cost-preserving transformations, we give a classification of nodes in P. The vertices of $\mathcal{T}$ that are nodes of P are called the nodes of P *inherited* from $\mathcal{T}$. The endpoints of degenerate links are called *degenerate nodes*. The other nodes are classified as follows. (Figure 3 shows some examples.)

- *Transversal node* : $p_i \in \text{int}(e)$ for some edge e, and $\overline{p_{i-1}p_i} \cap \text{int}(f) \neq \emptyset$ and $\overline{p_i p_{i+1}} \cap \text{int}(g) \neq \emptyset$, where f and g are the two faces incident to e.
- *Critical node of entry* : $p_i \in \text{int}(e)$ and $\text{int}(\overline{p_i p_{i+1}}) \cap e \neq \emptyset$ for some edge e and $\overline{p_{i-1}p_i} \cap \text{int}(f) \neq \emptyset$ for a face f incident to e.
- *Critical node of exit* : $p_i \in \text{int}(e)$ and $\text{int}(\overline{p_{i-1}p_i}) \cap e \neq \emptyset$ for some edge e, and $\overline{p_i p_{i+1}} \cap \text{int}(f) \neq \emptyset$ for a face f incident to e.
- *Reflective node*: $p_i \in \text{int}(e)$ for some edge e, and $\overline{p_{i-1}p_i} \cap \text{int}(f) \neq \emptyset$ and $\overline{p_i p_{i+1}} \cap \text{int}(f) \neq \emptyset$ for a face f incident to e.
- *Interior node* : $p_i \in \text{int}(f)$ for some face f.
- *Linear node*: $p_i \in \text{int}(e)$, $\text{int}(\overline{p_{i-1}p_i}) \cap e \neq \emptyset$ and $\text{int}(\overline{p_i p_{i+1}}) \cap e \neq \emptyset$ for some edge e.

In the definition of a linear, interior or reflective node p_i, we allow $\overline{p_{i-1}p_i}$ and $\overline{p_i p_{i+1}}$ to overlap. We call a polygonal path *non-redundant* if it does not contain any degenerate, reflective, interior, or linear node. Some intermediate results of the cost-preserving transformations may be self-intersecting paths, so our classification of nodes does not assume simplicity. Our classification does not assume paths to be $\mathcal{T}$-respecting either.

Our cost-preserving transformations manipulate polygonal paths by modifying sub-paths and concatenating polygonal paths. Recall that, for convenience, we use the same notation for a polygonal path and its sequence of nodes. For any integers i, j such that $0 \leqslant i \leqslant j \leqslant m$, we use $P[i, j]$ to denote the sub-path $(p_i, \dots, p_j)$. Given two sub-paths $A = (a_1, \dots, a_{m'})$ and $B = (b_1, \dots, b_{m''})$, we denote by $A \cdot B = (a_1, \dots, a_{m'}, b_1, \dots, b_{m''})$ the concatenation of A and B.

4.2 Splitting a Path

Let $P = (p_0, \dots, p_m)$ be a polygonal path. The procedure Split converts P into a $\mathcal{T}$-respecting path with the same cost. This procedure operates as follows. For each vertex v that lies in the interior of a link of P, it splits this link at v and makes v a node of P. For each crossing point x between an edge of $\mathcal{T}$ and a link of P, it splits the link at point x, and introduces the node x.

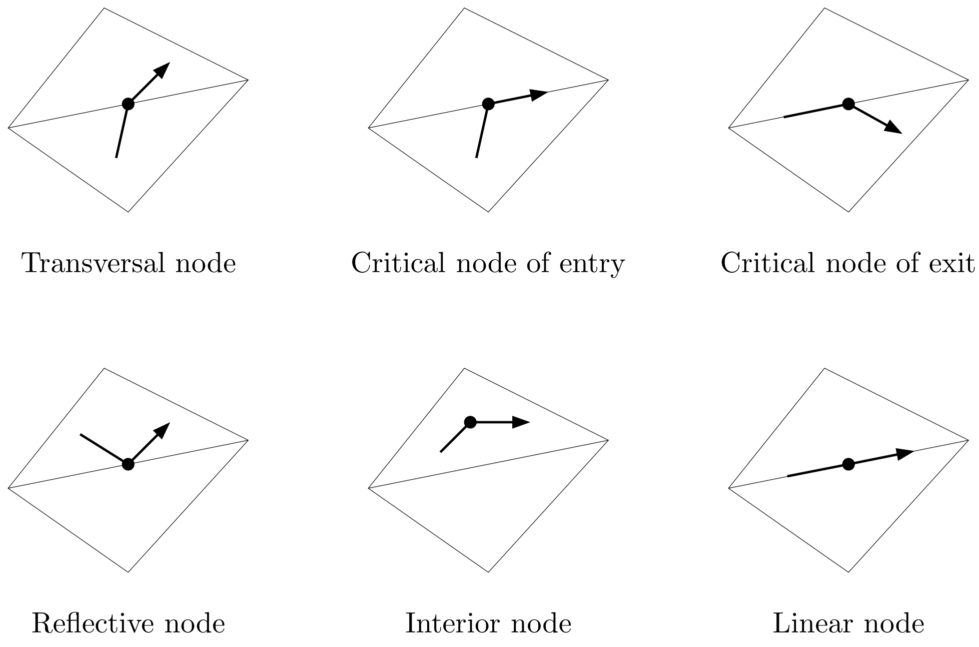

Figure 3: Different types of nodes of P other than degenerate nodes and nodes inherited from $\mathcal{T}$.

```
Split(Polygonal path $P$)

  1. If there exist a vertex $v$ of $\mathcal{T}$ and a link $\overline{p_i p_{i+1}}$ of $P$ such that $v \in \mathrm{int}(\overline{p_i p_{i+1}})$,
     then return Split($P[0,i] \cdot (v) \cdot P[i+1,m]$).
  2. If there exist an edge $e$ of $\mathcal{T}$, a link $\overline{p_i p_{i+1}}$ of $P$ and a point $x$ such that
     $x = \mathrm{int}(e) \cap \mathrm{int}(\overline{p_i p_{i+1}})$, then return Split($P[0,i] \cdot (x) \cdot P[i+1,m]$).
  3. Return $P$.
```

The following lemma describes the effect of Split.

Lemma 4.1 *For any polygonal path P, the path P' returned by* Split(P) *is $\mathcal{T}$-respecting and* $\mathrm{cost}(P') = \mathrm{cost}(P)$.

4.3 Reducing a Path

Let $P = (p_0, \ldots, p_m)$ be a $\mathcal{T}$-respecting path. The procedure Reduce eliminates some locally non-optimal features in P, including degenerate, reflective, interior, and linear nodes, as well as self-intersections.

```
Reduce($\mathcal{T}$-respecting path $P = (p_0, \ldots, p_m)$)

  1. If there exists $i$ such that $p_i = p_{i+1}$,
     then return Reduce($P[0, i-1] \cdot P[i+1, m]$).
  2. If a node $p_i$ is a reflective, interior, or linear node,
     then return Reduce($P[0, i-1] \cdot P[i+1, m]$).
```

```
3. If $P$ self-intersects, then we are in one of the following two cases:
    (a) There exist $i < j$ and $x \in |\mathcal{T}|$ such that $x \in \overline{p_{i-1}p_i} \cap \overline{p_j p_{j+1}}$.
        Then return $\mathsf{Reduce}(P[0, i-1] \cdot (x) \cdot P[j+1, m])$.
    (b) There exists $i$ such that $p_{i-1} \in \overline{p_i p_{i+1}}$ or $p_{i+1} \in \overline{p_{i-1}p_i}$.
        Then return $\mathsf{Reduce}(P[0, i-1] \cdot P[i+1, m])$.
4. Return $P$.
```

It is clear that the number of nodes in P decreases after one application of step 1, 2 or 3. It implies that Reduce terminates.

Recall that the input path is assumed to be $\mathcal{T}$-respecting. Clearly, this condition continues to hold after applying step 1, 2 or 3 any number of times. In addition, when Reduce terminates, the output path P' returned by Reduce(P) is simple and non-redundant.

Lemma 4.2 *Let P be a $\mathcal{T}$-respecting path. The path P' returned by* Reduce(P) *is simple, non-redundant, and $\mathcal{T}$-respecting. If $P' \neq P$, then P' has fewer links than P and* $\text{cost}(P') \leqslant \text{cost}(P)$.

4.4 Sliding Sub-paths

We call (i, j) a *critical pair* if p_i is a critical node of exit, p_j is a critical node of entry, $i < j$, and all of the nodes in the interior of the sub-path $P[i, j]$ are transversal nodes. (See Figure 4.) We introduce a procedure Slide to remove a critical pair in P, if there is one. The same technique of sliding sub-paths was used by Mitchell and Papadimitriou [13]. Because Reduce will be applied before applying Slide, we assume that P is a simple, non-redundant, $\mathcal{T}$-respecting path.

```
Slide(simple, non-redundant, $\mathcal{T}$-respecting path $P$)
    1. Look for a critical pair $(i, j)$ in $P$.
    2. If no critical pair is found, return $P$.
    3. Slide $P[i, j]$ in a direction such that $\text{cost}(P)$ does not increase, until one of the
       following situations occurs:
        (a) $p_{i-1} = p_i$. Then return $P[0, i-1] \cdot P[i+1, m]$.
        (b) $p_j = p_{j+1}$. Then return $P[0, j-1] \cdot P[j+1, m]$.
        (c) $P$ self-intersects or $P[i, j]$ hits a vertex of $\mathcal{T}$. Then return $P$.
```

We explain the sliding operation in step 3 in more detail. Refer to Figure 4. Sliding $P[i, j]$ means shifting p_i, all of the transversal nodes on $P[i, j]$, and p_j along the corresponding edges in such a way that the links in $P[i, j]$ remain parallel to their original positions. The links $\overline{p_{i-1}p_i}$, $\overline{p_j p_{j+1}}$, and those in $P[i, j]$ may lengthen or shrink. Let Δ be the signed distance of the sliding of p_i from its original position. We take Δ to be positive (resp. negative) if p_i slides to the right (resp. left). Take a sliding link $\overline{p_a p_{a+1}}$. As the slope of $\overline{p_a p_{a+1}}$ is kept constant, the cost of $\overline{p_a p_{a+1}}$ is an affine function of Δ. Clearly, the costs of $\overline{p_{i-1}p_i}$ and $\overline{p_j p_{j+1}}$ are also affine functions of Δ. Thus, the total change in $\text{cost}(P)$ is an affine function of Δ. Since the value of an affine function is minimized at the boundary of its domain, we conclude that there is a direction in which we can slide $P[i, j]$ without increasing $\text{cost}(P)$. The stopping criteria in step 3 exhausts all the possibilities in which (i, j) no longer satisfies the criteria of being a critical pair or P ceases to be simple. The following lemma describes the effect of Slide.

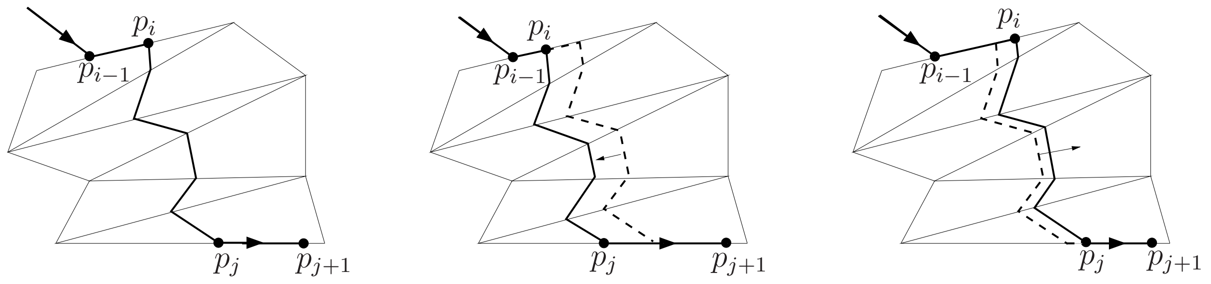

Figure 4: A critical pair (i, j). The sub-path $P[i, j]$ can slide to the left or to the right.

Lemma 4.3 *Let P be a simple, non-redundant, $\mathcal{T}$-respecting path. The path P' returned by* $\mathsf{Slide}(P)$ *satisfies the following properties:*

(i) *P' is a $\mathcal{T}$-respecting path.*

(ii) $\text{cost}(P') \leqslant \text{cost}(P)$ *and P' has no more links than P.*

(iii) *If $P' \neq P$ and P' has as many links as P, then P' is self-intersecting or P' has more nodes inherited from $\mathcal{T}$ than P does.*

4.5 Combining the Transformations

We define a procedure Simplify that makes use of the previous procedures. The input is a polygonal path P.

```
Simplify(Polygonal path $P$)
    1. $Q_1 := \mathsf{Split}(P)$.
    2. $Q_2 := \mathsf{Reduce}(Q_1)$.
    3. $Q_0 := \mathsf{Slide}(Q_2)$. If $Q_0 \neq Q_2$, set $Q_1 := Q_0$ and go back to step 2.
    4. Return $Q_0$.
```

Lemma 4.4 *Let P be a polygonal path. The path Q_0 returned by* $\mathsf{Simplify}(P)$ *has the following properties:*

(i) $\text{cost}(Q_0) \leqslant \text{cost}(P)$.

(ii) *Q_0 is a simple, non-redundant, $\mathcal{T}$-respecting path.*

(iii) *Q_0 has no critical pair and has at most $2n$ critical vertices of entry or exit.*

Proof. We first show the termination of Simplify. By Lemmas 4.2 and 4.3, the number of links in the path never increases at steps 2 and 3. If the number of links stays the same in step 3, then Lemma 4.3 says that Q_0 is self-intersecting or it has more nodes inherited from $\mathcal{T}$ than Q_2 does. If Q_0 is self-intersecting, the number of links in the path decreases when Reduce is called in the next step. So if steps 2 and 3 iterate without decreasing the number of links in the path,

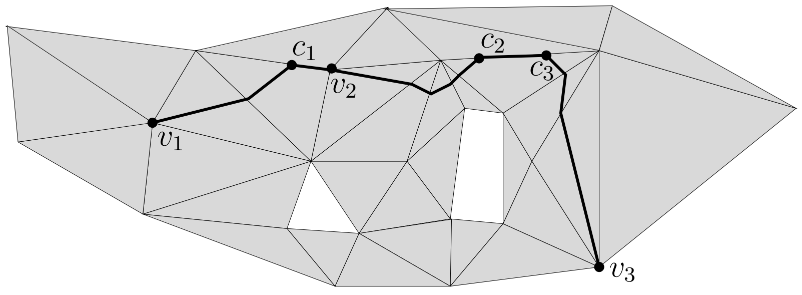

Figure 5: An example of a simplified path Q_0. There are two critical nodes c_2 and c_3 between vertices v_2 and v_3.

it means that the path remains simple and the number of nodes inherited from $\mathcal{T}$ increases from iteration to iteration. But this can happen at most $n-2$ times before all vertices of $\mathcal{T}$ lie on the path. Therefore, the number of links in the path must decrease after at most $n-1$ iterations of steps 2 and 3. Clearly, the number of links cannot decrease below one, so Simplify must terminate.

Let $(q_0, \ldots, q_m)$ be the node sequence of Q_0. Property (i) follows from the fact that Split, Reduce and Slide do not increase the cost. Property (ii) follows from Lemma 4.2. The termination of Simplify implies that no critical pair is detected by Slide. So if we walk along Q_0 from v_s to v_d and encounter a critical node of exit q_i, we will see a node $q_{i'}$ inherited from $\mathcal{T}$ before encountering any critical node of entry or exit. We charge q_i to $q_{i'}$. The node $q_{i'}$ can only be charged once, so there are at most n critical nodes of exit in Q_0. By a symmetric argument (following Q_0 backward from v_d to v_s), we can show that there are at most n critical nodes of entry. Hence, the total number of critical nodes is at most $2n$. ◻

In fact, the proof of Lemma 4.4 shows even more. Between any two consecutive nodes of Q_0 inherited from $\mathcal{T}$, there is at most one critical node of entry and one critical node of exit. In case these two critical nodes are present, they lie on the same edge e, and the path enters and exits e from the same side. (See Figure 5.) All of the other nodes (between these two consecutive inherited nodes) are transversal nodes.

5 Path Complexity

In this section we show that any polygonal path P can be approximated by an $(21\rho n^2/\varepsilon)$-link path with cost at most $(1+\varepsilon)\operatorname{cost}(P)$. This result, combined with the algorithms for approximating k-link shortest paths that we presented in Section 3, allows us to compute a polygonal path with cost at most $(1+\varepsilon)\operatorname{cost}(P)$ in time $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$.

5.1 Shortcut near Vertices

We introduce a procedure Shortcut that removes nodes in the vicinity of vertices of $\mathcal{T}$. Shortcut takes as input a polygonal path P, a real number $\delta > 0$, and a vertex v of $\mathcal{T}$. Shortcut will be invoked repeatedly, starting with the output of Simplify. The path may no longer be simple or non-redundant after several applications of Shortcut.

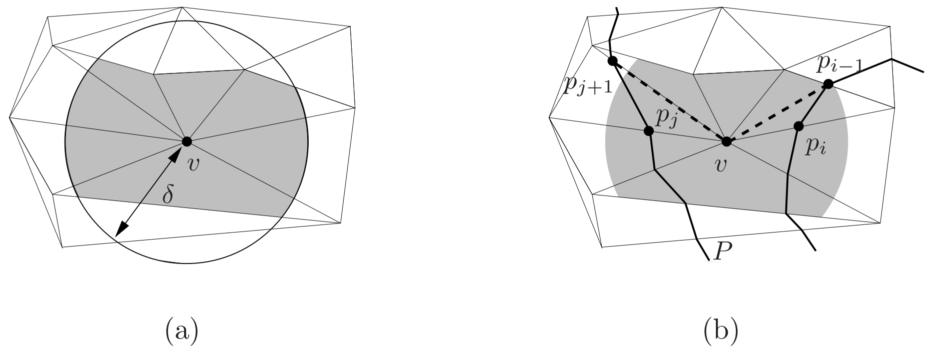

Figure 6: (a) The δ-neighborhood of v is shaded. (b) The effect of Shortcut(P, δ, v). The subpath $P[i-1, j+1]$ is replaced by the dashed subpath (p_{i-1}, v, p_{j+1}).

Let v be a vertex of $\mathcal{T}$. The *δ–neighborhood* of v is the intersection of the δ-radius Euclidean disk centered at v with the union of the faces that contain v. (See Figure 6a.) Shortcut(P, δ, v) checks if P has a node in the interior of the δ–neighborhood of v. If so, let p_i and p_j be the first and last nodes of P, respectively, in the interior of the δ-neighborhood of v—here the interior is taken in the topological sense. Then Shortcut replaces $P[i-1, j+1]$ with the sub-path (p_{i-1}, v, p_{j+1}). (See Figure 6b.) The nodes p_{i-1} and p_{j+1} lie on the boundary of a face incident to v, so the new path is still inside $|\mathcal{T}|$. The links $\overline{p_{i-1}v}$ and $\overline{vp_{j+1}}$ introduced by Shortcut may make the path redundant or self-intersecting. Here is the pseudo-code of Shortcut.

```
Shortcut(Polygonal path $P$, $\delta > 0$ , vertex $v$)
    1. If no node of $P$ is in the interior of the $\delta$–neighborhood of $v$, return $P$.
    2. Let $p_i$ and $p_j$ be the first and last nodes along $P$, respectively, that lie in the
       interior of the $\delta$–neighborhood of $v$. Let $p_0$ and $p_m$ be the first and last nodes
       of $P$.
    3. If $v = p_0$, return $(v) \cdot P[j+1, m]$.
    4. If $v = p_m$, return $P[0, i-1] \cdot (v)$.
    5. Return $P[0, i-1] \cdot (v) \cdot P[j+1, m]$.
```

The following lemma gives a bound on the cost of the path obtained by applying Shortcut.

Lemma 5.1 *Let P be a $\mathcal{T}$-respecting path. The path returned by* Shortcut(P, δ, v) *has cost less than* $\text{cost}(P) + 2\rho\delta$.

Proof. Either $\overline{p_{i-1}p_i}$ is a chord of a face incident to v, or it is contained in an edge incident to v. In any case, as $\|p_i v\| < \delta$, we have $\text{cost}(\overline{p_{i-1}v}) < \text{cost}(\overline{p_{i-1}p_i}) + \rho\delta$. Similarly, we can prove that $\text{cost}(\overline{vp_{j+1}}) < \text{cost}(\overline{p_j p_{j+1}}) + \rho\delta$. ◻

Remark: Reif and Sun [19] also used a disk neighborhood. Their disk radius is related to the local geometry around the vertex v and the disk is completely contained in the union of the faces incident to v. In our construction, the disk radius will be set to be proportional to the global path cost. (See the procedure Convert in the next section.) As shown in Figure 6, our disk may not be contained in the union of faces incident to v.

5.2 Path Conversion

The pseudo-code below describes a procedure Convert that transforms a polygonal path P into another polygonal path R. We show that R has $21\rho n^2/\varepsilon$ links and its cost is at most $(1+\varepsilon)\,\mathrm{cost}(P)$. We denote by $\{v_1, v_2, \ldots, v_n\}$ the vertices of $\mathcal{T}$ in the following pseudo-code.

```
Convert(Polygonal path $P$, $\varepsilon \in (0,1)$)
  1. $Q_0 :=$ Simplify($P$).
  2. $\delta := \varepsilon\,\mathrm{cost}(P)/(2\rho n)$.
  3. For $i = 1$ to $n$, $Q_i :=$ Shortcut($Q_{i-1}, \delta, v_i$).
  4. $R := Q_n$.
  5. Return $R$.
```

The procedure Convert has the following properties.

Lemma 5.2 *Let $R = (r_1, \ldots, r_\ell)$ be the path returned by* Convert(P, ε)*.*

(i) $\mathrm{cost}(R) \leqslant (1+\varepsilon)\ \mathrm{cost}(P)$.

(ii) *R is a $\mathcal{T}$-respecting path with distinct nodes.*

(iii) *If $\{r_i, r_{i+1}\}$ contains no vertex of $\mathcal{T}$, then $\overline{r_i r_{i+1}}$ is a link of Q_0.*

(iv) *If v is a vertex of $\mathcal{T}$ and r_i lies in the interior of an edge incident to v, then $\|r_i v\| \geqslant \delta$.*

Proof. By Lemma 5.1, the path cost increases by at most $2\rho\delta = \varepsilon\,\mathrm{cost}(P)/n$ for each call to Shortcut. By Lemma 4.4, $\mathrm{cost}(Q_0) \leqslant \mathrm{cost}(P)$, thus $\mathrm{cost}(R) \leqslant \mathrm{cost}(Q_0) + \varepsilon\ \mathrm{cost}(P) \leqslant (1+\varepsilon)\ \mathrm{cost}(P)$. By Lemma 4.4 again, Q_0 is a $\mathcal{T}$-respecting path with distinct nodes. Each call to Shortcut preserves these two properties. It proves (ii). Properties (iii) and (iv) follow from the working of Shortcut. ◻

We prove a technical lemma which essentially says that, for any edge e, the transversal nodes in the output of Convert(P, ε) that lie on e are sparse.

Lemma 5.3 *Let $R = (r_1, \ldots, r_\ell)$ be the path returned by* Convert(P, ε)*. Consider an edge e of $\mathcal{T}$ and a node $r_i \in \mathrm{int}(e)$. Suppose that the following conditions hold:*

- *There exists a node $r_j \in \mathrm{int}(e)$ with $j > i+1$. If there are several such nodes we choose j to be the minimum.*
- *All nodes in $\mathrm{int}(R[i,j])$ are transversal nodes.*

Then $\mathrm{cost}(R[i,j]) > \delta$.

Proof. We denote by ℓ_e the support line of e, and we denote by x the first intersection point between $R[i,j]$ and ℓ_e. If $x \neq r_j$, then $x \notin \mathrm{int}(e)$ as R is $\mathcal{T}$-respecting. (See Figure 7a.) By Lemma 5.2(iv), we know that $\|r_i x\| > \delta$ and so $\mathrm{cost}(R[i,j]) \geqslant \|r_i x\| > \delta$.

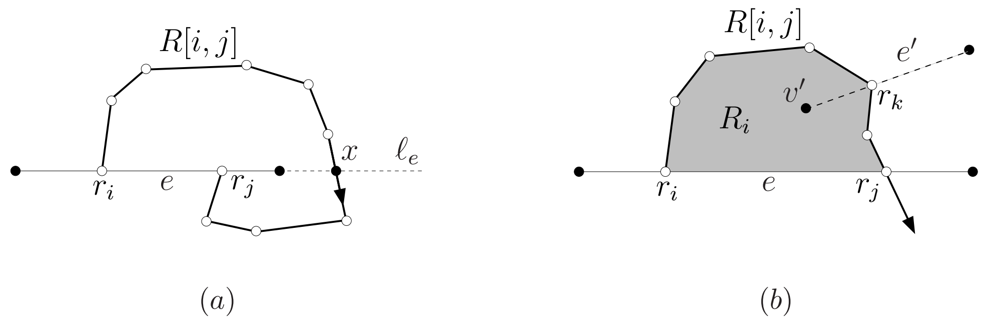

Figure 7: In (a), the sub-path $R[i,j]$ intersects ℓ_e outside e at point x. In (b), the node v' is in the interior of the region R_i delimited by $R[i,j]$ and $\overline{r_i r_j}$.

We now assume that $x = r_j$, and thus $\text{int}(R[i,j])$ lies entirely on one side of ℓ_e. We denote by R_i the region delimited by $R[i,j]$ and $\overline{r_i r_j}$. (See Figure 7b.) We first prove that there exists an edge e' of $\mathcal{T}$ with one endpoint v' inside R_i. Let E denote the set of edges of $\mathcal{T}$ that contain a node of $R[i,j]$. Pick an edge of E. If this edge has an endpoint inside R_i, we are done. Otherwise, this edge crosses $\text{int}(R[i,j])$ transversally, so it separates a portion of R_i away from e. By recursively applying the argument on this portion of R_i, we must find an edge $e' \in E$ that has an endpoint inside R_i.

Let r_k denote a node of $R[i,j]$ that lies on e'. Because no vertex of $\mathcal{T}$ lies on $R[i,j]$, Lemma 5.2(iv) implies that $\|r_k v'\| \geqslant \delta$. It follows that $\text{length}(R[i,j]) > \|r_k v'\| \geqslant \delta$. Hence, $\text{cost}(R[i,j]) \geqslant \text{length}(R[i,j]) > \delta$. ⊔⊓

We are now ready to derive a pseudo-polynomial bound on the path complexity.

Theorem 2 *For any* $\varepsilon \in (0,1)$ *and for any polygonal path* P*, there exists a path* P^ε *with at most* $21\rho n^2/\varepsilon$ *links such that* $\text{cost}(P^\varepsilon) \leqslant (1+\varepsilon)\,\text{cost}(P)$*. In addition,* P^ε *can be chosen to be simple and non-redundant.*

Proof. Let $Q_0 = \mathsf{Simplify}(P)$, $R = \mathsf{Convert}(P,\varepsilon)$, and $P^\varepsilon = \mathsf{Reduce}(R)$. By Lemmas 4.2, 4.4, and 5.2, we know that:

- P^ε is simple, non-redundant, and $\mathcal{T}$-respecting.
- $\text{cost}(P^\varepsilon) \leqslant \text{cost}(R) \leqslant (1+\varepsilon)\ \text{cost}(P)$.

Reduce does not increase the number of links. Hence, it suffices to prove that R has $21\rho n^2/\varepsilon$ nodes. By Lemma 5.2(ii), R is a $\mathcal{T}$-respecting path with distinct nodes.

Take an edge e of $\mathcal{T}$. We denote by N_e the set of nodes $r_i \in \text{int}(e)$. We show below that $|N_e| \leqslant 4\rho n/\varepsilon + 3n + 4$. We assume that $|N_e| \geqslant 2$; otherwise, we are done. Let r_i and r_j, $i < j$, be two nodes in N_e that are consecutive in the order along R.

Case 1: r_{i-1} is a vertex of $\mathcal{T}$. Because $r_i \in \text{int}(e)$, r_{i-1} must be one of the four vertices of the two triangles incident to e. So there are at most four such r_i's in N_e.

Case 2: There is a vertex v of $\mathcal{T}$ in $R[i,j]$. We charge r_i to v, thus there are at most n such r_i's in N_e.

Case 3: $j = i + 1$. As $r_i, r_{i+1} \in \operatorname{int}(e)$, r_i and r_{i+1} are not vertices of $\mathcal{T}$. Neither is r_{i-1} as case 1 does not apply. By Lemma 5.2(iii), (r_{i-1}, r_i, r_{i+1}) is a subpath of Q_0. We have $\overline{r_i r_{i+1}} \subset \operatorname{int}(e)$ as $r_i, r_{i+1} \in \operatorname{int}(e)$. It follows that r_i is a critical node in Q_0 because Q_0 is non-redundant and $\mathcal{T}$-respecting.

Case 4: There is no vertex of $\mathcal{T}$ in $R[i,j]$, but there is a critical node r_k in $\operatorname{int}(R[i,j])$. We charge r_i to r_k. Lemma 5.2(iii) implies that $R[i,j]$ is a subpath of Q_0. So r_k is also a critical node in Q_0.

Case 5: There is no vertex of $\mathcal{T}$ in $R[i,j]$ and no critical node in $\operatorname{int}(R[i,j])$. Again, Lemma 5.2(iii) implies that $R[i,j]$ is a subpath of Q_0. Because Q_0 is $\mathcal{T}$-respecting and non-redundant, all of the nodes in $\operatorname{int}(R[i,j])$ are transversal nodes. By Lemma 5.3, we have $\operatorname{cost}(R[i,j]) > \delta$. Because $\operatorname{cost}(R) \leqslant (1+\varepsilon)\operatorname{cost}(P) < 2\operatorname{cost}(P)$ and $\delta = \frac{\varepsilon}{2\rho n}\operatorname{cost}(P)$, we get $\operatorname{cost}(R[i,j]) > \frac{\varepsilon}{4\rho n}\operatorname{cost}(R)$. So there are at most $4\rho n/\varepsilon$ such r_i's in N_e.

Cases 1, 2 and 5 contribute at most $4\rho n/\varepsilon + n + 4$ nodes to N_e. The nodes in cases 3 and 4 are critical nodes in Q_0. By Lemma 4.4(iii), there are at most $2n$ critical nodes in Q_0, so cases 3 and 4 contribute at most $2n$ nodes to N_e. In all, $|N_e| \leqslant 4\rho n/\varepsilon + 3n + 4$. The number of edges is at most $3n - 6$ for a planar graph with n vertices. Summing over all edges of $\mathcal{T}$ and including the vertices of $\mathcal{T}$, the number of nodes in R is at most $n + (3n-6)(4\rho n/\varepsilon + 3n + 4) \leqslant 21\rho n^2/\varepsilon$. $\square$

6 General Paths

The statement of Theorem 2 is not entirely satisfactory. For instance, it does not tell us whether there exists a shortest path and if so, whether it is a polygonal path. We give an example in Section 6.1 in which the shortest path cannot be polygonal. Nonetheless, we show that there exists a shortest rectifiable path. Intuitively, a path is rectifiable if its length is finite; for example, the class of rectifiable paths includes piecewise C^1 paths. Furthermore, we show that for any $\alpha > 1$, there exists a polygonal path with cost at most α times the optimal. These results imply Corollary 2 in Section 3.4.

Throughout this section we do not require the endpoints of a path to be v_s and v_d. Also, we do not restrict paths to be polygonal unless stated explicitly otherwise.

6.1 Example of a Non-polygonal Shortest Path

There may not be any (exact) shortest polygonal path. We give an example in Figure 8 in which $\rho = \sqrt{2}$ and there is no shortest polygonal path.

We prove it by contradiction. Consider the points x and c in Figure 8. Assume that the last link of a polygonal path P is $\overline{xv_d}$. Then, if we insert the node c between x and v_d in P (thus replacing the sub-path (x, v_d) by (x, c, v_d)), we obtain a polygonal path with strictly smaller cost than P. The same argument shows that the shortest path intersects some edge an infinite number of times.

A similar example can be constructed for any $\rho > 1$, by changing B_{f_i} into a regular polygon with enough edges and by placing a larger number of faces around v_d.

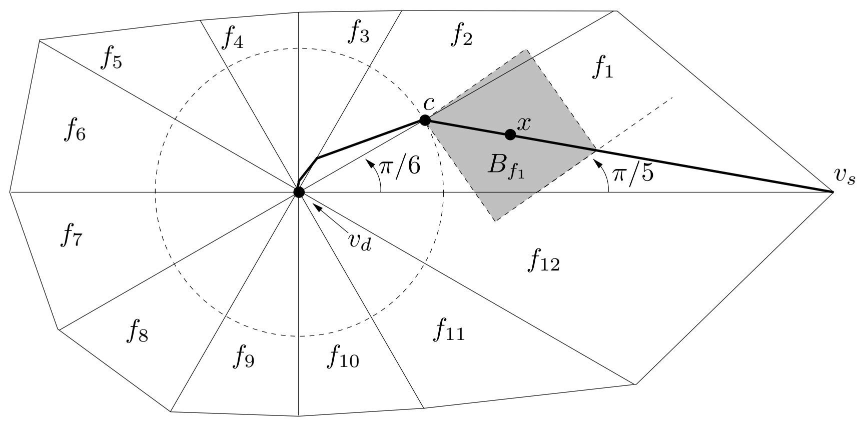

Figure 8: B_{f_1} is a square centered at the origin and with edge length $\sqrt{2}$. B_{f_i} is obtained by rotating B_{f_1} by an angle $(i-1)\pi/6$. We know that the shortest path from v_s to v_d is not polygonal. We conjecture that the shortest path from v_s to v_d is an infinite sequence of line segments, forming a spiral around v_d.

6.2 Rectifiable Paths

Intuitively, a path is rectifiable if it has finite length. The length of a path can be defined as the supremum of the length of the polygonal paths inscribed in this path. This definition is a common way of introducing the length of a curve. (For instance, see Guggenheimer's book [9].) It allows one to define the length of a curve that is not necessarily piecewise C^1. It can also be proved that, for a C^1 curve, the length defined in this way coincides with the other usual definition through calculus, where the length is obtained by integrating the norm of the derivative of the curve [9, Theorem 2-3].

We do not restrict paths to be polygonal, so a path is a continuous function $\pi : [a,b] \to |\mathcal{T}|$ where $a < b \in \mathbb{R}$. We use $\pi[a,b]$ to denote $\{\pi(t) : a \leqslant t \leqslant b\}$ and $\pi(a,b)$ to denote $\{\pi(t) : a < t < b\}$. We say that π is a path from x to y if $\pi(a) = x$ and $\pi(b) = y$.

An $[a,b]$-*sequence* is a finite increasing sequence $\sigma = (t_0, t_1, \ldots, t_m)$ such that $a = t_0 < t_1 < \cdots < t_m = b$. A polygonal path *inscribed in* π is a polyline $\pi_\sigma = (\pi(t_0), \pi(t_1), \ldots, \pi(t_m))$ such that σ is an $[a,b]$-sequence for some $a < b$. When there is no ambiguity, we abuse notation and use π_σ to denote $\bigcup_{i=1}^{m} \overline{\pi(t_{i-1})\pi(t_i)}$.

The *length* of a path $\pi : [a,b] \to |\mathcal{T}|$ is defined as

$$\text{length}(\pi) = \sup\{\,\text{length}(\pi_\sigma) : \sigma \text{ is an } [a,b]\text{-sequence}\,\}.$$

The path π is *rectifiable* if length(π) is finite.

An inscribed polygonal path π_σ is not necessarily contained in $|\mathcal{T}|$. However, there always exists an inscribed polygonal path contained in $|\mathcal{T}|$:

Lemma 6.1 *For any path $\pi : [a,b] \to |\mathcal{T}|$, there exists an $[a,b]$-sequence σ such that $\pi_\sigma \subset |\mathcal{T}|$.*

Proof. Let F denote the set of edges e of $\mathcal{T}$ such that $e \cap \pi(a,b) \neq \emptyset$. We prove the lemma by induction on $|F|$. If $F = \emptyset$, then $\pi(a,b)$ lies in a face of $\mathcal{T}$ and we are done. Otherwise, let

S denote the union of the faces that contain $\pi(a)$. Note that S is star-shaped around a, i.e., for any $x \in S$, we have $\overline{ax} \subset S$. If $\pi(b) \in S$, we are done (choose $\sigma = (a, b)$). Otherwise, let $c = \max\{t : \pi(t) \in S\}$. Then $\pi(c, b)$ does not intersect any edge e in F such that $e \subset S$. So, by our induction hypothesis, there is a $[c, b]$-sequence σ' such that $\pi_{\sigma'} \subset |\mathcal{T}|$. Choosing σ as the concatenation of (a) and σ', we conclude that $\pi_\sigma \subset |\mathcal{T}|$. □

6.3 Cost Distance

We introduce a new distance function and prove some of its properties. This distance function yields a measure of the cost of a path. We prove that this new cost measure coincides with the path cost defined in Section 2 in the case of polygonal paths. We also prove the existence of an optimal rectifiable path under this cost measure, and so the optimal path can be approximated using our algorithmic result in Section 3.

Let x, y be two points in $|\mathcal{T}|$. The *cost distance* from x to y, denoted by $\mathrm{d}(x, y)$, is defined as follows.

$$\mathrm{d}(x, y) = \inf\{\mathrm{cost}(P) : P \text{ is a polygonal path from } x \text{ to } y \text{ and } P \subset |\mathcal{T}|\}.$$

The distance function $\mathrm{d}(\cdot, \cdot)$ is well defined, as the cost of a polygonal path is well defined and positive. It has several useful properties, listed in the following lemma. In particular, it is continuous. On the contrary, $\mathrm{cost}(\overline{xy})$ may not be continuous in (x, y); for instance, when we move $\overline{xy}$ from the interior of a face to its boundary.

Lemma 6.2 *The cost distance has the following properties.*

(i) *For any $x, y \in |\mathcal{T}|$, $x = y$ if and only if $\mathrm{d}(x, y) = 0$.*

(ii) *For any $x, y, z \in |\mathcal{T}|$, $\mathrm{d}(x, z) \leqslant \mathrm{d}(x, y) + \mathrm{d}(y, z)$.*

(iii) *For any $x, y \in |\mathcal{T}|$, $\|xy\| \leqslant \mathrm{d}(x, y)$.*

(iv) *$\mathrm{d}(\cdot, \cdot)$ is continuous over $|\mathcal{T}|^2$.*

(v) *For any $x, z \in |\mathcal{T}|$, there exists $y \in |\mathcal{T}|$ such that $\mathrm{d}(x, y) = \mathrm{d}(y, z) = \mathrm{d}(x, z)/2$.*

Proof. The correctness of (i), (ii) and (iii) follow from the definition of $\mathrm{d}(\cdot, \cdot)$ and the triangle inequality.

We now prove (iv). Let x_0 and y_0 be two points in $|\mathcal{T}|$. Let D_δ and D'_δ denote the disks centered at x_0 and y_0 with radius δ, respectively. We assume that $\delta > 0$ is sufficiently small so that $D_\delta \cap |\mathcal{T}|$ and $D'_\delta \cap |\mathcal{T}|$ are star-shaped around x_0 and y_0, respectively. So, for any $x \in D_\delta \cap |\mathcal{T}|$ and for any $y \in D'_\delta \cap |\mathcal{T}|$, we have $|\mathrm{d}(x, y) - \mathrm{d}(x_0, y_0)| \leqslant \mathrm{d}(x, x_0) + \mathrm{d}(y_0, y) \leqslant \rho \|x_0 x\| + \rho \|y_0 y\|$. It follows that $\lim_{(x,y)\to(x_0,y_0)} \mathrm{d}(x, y) = \mathrm{d}(x_0, y_0)$. Thus $\mathrm{d}(\cdot, \cdot)$ is continuous over $|\mathcal{T}|^2$.

We now prove (v). By definition, for any $n \in \mathbb{N}$, there exists a polygonal path $A_n \subset |\mathcal{T}|$ from x to z such that $\mathrm{cost}(A_n) \leqslant \mathrm{d}(x, z) + 2/n$. There is a point y_n on A_n such that the cost of the sub-path from x to y_n is $\mathrm{cost}(A_n)/2$. Then $\mathrm{d}(x, y_n) \leqslant \mathrm{d}(x, z)/2 + 1/n$ and $\mathrm{d}(y_n, z) \leqslant \mathrm{d}(x, z)/2 + 1/n$. Since $|\mathcal{T}|$ is compact, $\{y_n\}$ has a limit point, which we denote by y. By (iv), the cost distance $\mathrm{d}(\cdot, \cdot)$ is continuous, so $\mathrm{d}(x, y) \leqslant \lim_{n\to\infty} \mathrm{d}(x, z)/2 + 1/n = \mathrm{d}(x, z)/2$.

Similarly, $d(y,z) \leqslant d(x,z)/2$. We complete the proof using (ii). $\square$

Properties (i) and (ii) show that $d(\cdot,\cdot)$ is a quasi-metric: it is similar to a metric, except that it is not symmetric.

6.4 Cost of a Rectifiable Path

We define a measure of path cost using the distance function $d(\cdot,\cdot)$. Our definition is similar to the definition of the length of a rectifiable path. The exposition of this section follows the lecture notes by Lang [10] on metric geometry, and the book by Burago et al. [3]. The results in these notes and this book do not apply to our problem directly, because $d(\cdot,\cdot)$ is not a metric, so we reprove the results we need.

The d-*cost* of a path $\pi : [a,b] \to |\mathcal{T}|$ is defined as

$$C(\pi) = \sup\left\{ \sum_{i=1}^{m} d(\pi(t_{i-1}), \pi(t_i)) : \sigma = (t_0, \ldots, t_m) \text{ is an } [a,b]\text{-sequence such that } \pi_\sigma \subset |\mathcal{T}| \right\}.$$

By Lemma 6.1, the above supremum is well defined. We say that π is $\mathcal{T}$*-rectifiable* if $C(\pi)$ is finite.

The above definition of the d-cost works for paths specified as maps. We can extend it to polygonal paths as follows. Let $P = (p_0, \ldots, p_m)$ be a polygonal path. Construct the map $\pi : [0,1] \to |\mathcal{T}|$ such that $\pi(i/m) = p_i$ for any integer $i \in [0,m]$, and π is affine over $[(i-1)/m, i/m]$ for any integer $i \in [1,m]$. Then we define $C(P) = C(\pi)$. The following lemma shows that $C(P)$ coincides with $\mathrm{cost}(P)$ as defined in Section 2.

Lemma 6.3 *For any polygonal path P, we have $C(P) = \mathrm{cost}(P)$.*

Proof. We associate P with a function $\pi : [0,1] \to |\mathcal{T}|$ as explained earlier. Since $\pi[t,t']$ is a polygonal path from $\pi(t)$ to $\pi(t')$, by the definition of $d(\cdot,\cdot)$, we know that for any $t < t'$ in $[0,1]$, $d(\pi(t), \pi(t')) \leqslant \mathrm{cost}(\pi[t,t'])$. It follows that $C(P) \leqslant \mathrm{cost}(P)$.

We now prove that $C(P) \geqslant \mathrm{cost}(P)$. We put a ball with radius δ centered at each node of P, each crossing between P and the edges of $\mathcal{T}$, each crossing between links of P, and each vertex of $\mathcal{T}$ lying on P. We make δ sufficiently small so that the balls are mutually disjoint, each ball intersects only the link(s) of P that contain the ball center, and each ball intersects only the edge(s) and face(s) of $\mathcal{T}$ that contain the ball center.

Assume that there are k links in P. Since $\mathcal{T}$ has $O(n)$ vertices and edges, there are $O(k^2+kn)$ balls. We denote by $\mathcal{B}$ the set of points in these balls. Clearly, $P \setminus \mathcal{B}$ is a collection of disjoint line segments, each lying in the interior of an edge or a face of $\mathcal{T}$.

Pick a line segment $\overline{xy}$ in $P \setminus \mathcal{B}$. First, assume that $\overline{xy}$ lies in the interior of a face f of $\mathcal{T}$. Let ε be the Euclidean distance between $\overline{xy}$ and $\mathrm{bd}(f)$. Let $[a,b]$ be the subinterval of $[0,1]$ such that $\pi[a,b] = \overline{xy}$. Consider any $[a,b]$-sequence $(s_0, \ldots, s_\ell)$ such that $\|\overline{\pi(s_{i-1})\pi(s_i)}\| < \varepsilon/\rho$ for any i. So $\mathrm{cost}\left(\overline{\pi(s_{i-1})\pi(s_i)}\right) < \varepsilon$ for any i. Consider any polygonal path from $\pi(s_{i-1})$ to $\pi(s_i)$. If the path stays in $\mathrm{int}(f)$, its cost is no less than $\mathrm{cost}\left(\overline{\pi(s_{i-1})\pi(s_i)}\right)$. If the path reaches $\mathrm{bd}(f)$, then the path cost is at least $\varepsilon > \mathrm{cost}\left(\overline{\pi(s_{i-1})\pi(s_i)}\right)$. It follows that $d(\pi(s_{i-1}), \pi(s_i)) =$

cost $\left(\overline{\pi(s_{i-1})\pi(s_i)}\right)$ for any i. If $\overline{xy}$ lies in the interior of an edge of $\mathcal{T}$, let F be the union of the face(s) incident to e and let ε be the distance between $\overline{xy}$ and $\mathrm{bd}(F) \setminus \mathrm{int}(e)$. Again, we obtain $\mathrm{d}(\pi(s_{i-1}), \pi(s_i)) = \mathrm{cost}\left(\overline{\pi(s_{i-1})\pi(s_i)}\right)$ by considering separately polygonal paths that stay in $\mathrm{int}(F) \cup \mathrm{int}(e)$ and polygonal paths that reach $\mathrm{bd}(F) \setminus \mathrm{int}(e)$.

Consider any $[0,1]$-sequence $\sigma = (t_0, \ldots, t_m)$ such that $\pi_\sigma \subset |\mathcal{T}|$ and for each endpoint x of a segment in $P \setminus \mathcal{B}$, there exists $i \in [0,m]$ such that $\pi(t_i) = x$. This implies that if $[a,b]$ is a subinterval of $[0,1]$ such that $\pi[a,b] = \overline{xy}$ for a segment $\overline{xy}$ in $P \setminus \mathcal{B}$, then σ contains a $[a,b]$-sequence. Then the result in the previous paragraph implies that $\sum_{i=1}^m \mathrm{d}(\pi(t_{i-1}), \pi(t_i)) \geqslant \sum_{\overline{xy}} \mathrm{cost}(\overline{xy})$, where the second sum runs over all segment $\overline{xy}$ in $P \setminus \mathcal{B}$. Since $C(P) = C(\pi)$ is the supremum over all $[0,1]$-sequence that yields a polygonal path in $|\mathcal{T}|$, we conclude that $C(P) = C(\pi) \geqslant \sum_{\overline{xy}} \mathrm{cost}(\overline{xy})$.

The intersection $P \cap \mathcal{B}$ has a total length of $O(\delta(k^2 + kn))$. Therefore, $\mathrm{cost}(P \cap \mathcal{B}) \leqslant c\rho\delta(k^2 + kn)$ for some constant $c > 0$. By the triangle inequality, $\sum_{\overline{xy}} \mathrm{cost}\,\overline{xy} \geqslant \mathrm{cost}(P) - c\rho\delta(k^2 + kn)$, which tends to $\mathrm{cost}(P)$ as $\delta \to 0$. Hence, $C(P) \geqslant \mathrm{cost}(P)$. ◻

Consider the infimum of $C(\pi)$ over all rectifiable paths π from a point x to another point y in $|\mathcal{T}|$. Our goal is to show that some rectifiable path from x to y achieves this infimum and hence this path is shortest. First, we show that this infimum is equal to $\mathrm{d}(x,y)$.

Lemma 6.4 *For any $x, y \in |\mathcal{T}|$, $\mathrm{d}(x,y) = \inf\{C(\pi) : \pi$ is a rectifiable path from x to $y\}$.*

Proof. By definition, $\mathrm{d}(x,y)$ is the infimum of $\mathrm{cost}(P)$ over all polygonal paths P from x to y in $|\mathcal{T}|$. A polygonal path P is rectifiable and $C(P) = \mathrm{cost}(P)$ by Lemma 6.3. Therefore, $\mathrm{d}(x,y) \geqslant \inf\{C(\pi) : \pi$ is a rectifiable path from x to $y\}$. Assume to the contrary that $\mathrm{d}(x,y) > \inf\{C(\pi) :\ \pi$ is a rectifiable path from x to $y\}$. Then there exists a rectifiable path $\pi' : [a,b] \to |\mathcal{T}|$ from x to y such that $\mathrm{d}(x,y) > C(\pi')$. It follows from the definition of $C(\pi')$ that there exists an $[a,b]$-sequence $\sigma = (t_0, \ldots, t_m)$ such that

$$\mathrm{d}(x,y) > \sum_{i=1}^m \mathrm{d}(\pi'(t_{i-1}), \pi'(t_i)).$$

However, this is impossible by Lemma 6.2(ii). ◻

Next, we show that $\mathcal{T}$-rectifiability is equivalent to rectifiability. It follows that any rectifiable path has a finite d-cost (in particular, any piecewise C^1 path).

Lemma 6.5 *A path π in $|\mathcal{T}|$ is rectifiable if and only if π is $\mathcal{T}$-rectifiable.*

Proof. Suppose that a path $\pi : [a,b] \to |\mathcal{T}|$ is rectifiable. By definition, $C(\pi)$ is equal to the supremum of $\sum_{i=1}^m \mathrm{d}(\pi(t_{i-1}), \pi(t_i))$ over all $[a,b]$-sequences $\sigma = (t_0, \ldots, t_m)$ such that $\pi_\sigma \subset |\mathcal{T}|$. By Lemmas 6.3 and 6.4, we have $\mathrm{d}(\pi(t_{i-1}), \pi(t_i)) \leqslant C\left(\overline{\pi(t_{i-1})\pi(t_i)}\right) = \mathrm{cost}\left(\overline{\pi(t_{i-1})\pi(t_i)}\right)$. Thus, $C(\pi) \leqslant \sup_\sigma \mathrm{cost}(\pi_\sigma) \leqslant \sup_\sigma \rho\, \mathrm{length}(\pi_\sigma)$. By definition, $\mathrm{length}(\pi) \geqslant \mathrm{length}(\pi_\sigma)$. Therefore, $C(\pi) \leqslant \rho\, \mathrm{length}(\pi)$ which is finite as π is rectifiable. So π is $\mathcal{T}$-rectifiable.

Suppose that a path $\pi : [a,b] \to |\mathcal{T}|$ is $\mathcal{T}$-rectifiable. By definition, $\mathrm{length}(\pi) = \sup_\sigma \mathrm{length}(\pi_\sigma)$ where the supremum is taken over all $[a,b]$-sequences. For each such σ, by applying Lemma 6.1

to successive numbers in σ, we can find an $[a,b]$-sequence σ' such that σ is a subsequence of σ' (and thus $\text{length}(\pi_\sigma) \leqslant \text{length}(\pi_{\sigma'})$) and $\pi_{\sigma'} \subset |\mathcal{T}|$. Therefore, it is also true that $\text{length}(\pi) = \sup_\sigma \text{length}(\pi_\sigma)$ over all $[a,b]$-sequences σ such that $\pi_\sigma \subset |\mathcal{T}|$. By Lemma 6.2(iii), we have $\text{length}(\pi_\sigma) \leqslant \sum_{i=1}^m \mathrm{d}(\pi(t_{i-1}), \pi(t_i))$ for any $\sigma = (t_0, \ldots, t_m)$. Therefore, $\text{length}(\pi) = \sup_\sigma \text{length}(\pi_\sigma) \leqslant \sup_\sigma\{\sum_{i=1}^m \mathrm{d}(\pi(t_{i-1}), \pi(t_i))\}$ over all $[a,b]$-sequences $\sigma = (t_0, \ldots, t_m)$ such that $\pi_\sigma \subset |\mathcal{T}|$. By definition, the right hand side is equal to $C(\pi)$, which is finite as π is $\mathcal{T}$-rectifiable. So $\text{length}(\pi)$ is finite and π is rectifiable. ◻

We are ready to show that there exists a shortest rectifiable path from x to y. The proof is analogous to the proof of the midpoint lemma in the lecture notes by Lang [10] and the proof in the book by Burago et al. [3, Theorem 2.4.16].

Theorem 3 *For any $x, y \in |\mathcal{T}|$, there exists a rectifiable path π^* from x to y such that*

$$C(\pi^*) = \inf\{C(\pi) : \pi \textit{ is a rectifiable path from } x \textit{ to } y\}.$$

Proof. We claim that it suffices to prove the existence of a path $\pi^* : [0,1] \to |\mathcal{T}|$ from x to y such that $C(\pi^*) \leqslant \mathrm{d}(x,y)$. Notice that, if so, π^* is $\mathcal{T}$-rectifiable and hence rectifiable by Lemma 6.5. Then $\mathrm{d}(x,y) \leqslant C(\pi^*)$ by Lemma 6.4 and so $C(\pi^*) = d(x,y)$. We assume, without loss of generality, that $\mathrm{d}(x,y) = 1$.

Let $U = \{i/2^j : i, j \in \mathbb{N} \text{ and } 0 \leqslant i/2^j \leqslant 1\}$. First, we recursively define π^* over U, starting with $\pi^*(0) = x$ and $\pi^*(1) = y$. By Lemma 6.2(v), we can choose $\pi^*(1/2)$ such that $\mathrm{d}(\pi^*(0), \pi^*(1/2)) = \mathrm{d}(\pi^*(1/2), \pi^*(1)) = 1/2$. We repeat this process recursively: we choose $\pi^*(3/4)$ such that $\mathrm{d}(\pi^*(1/2), \pi^*(3/4)) = \mathrm{d}(\pi^*(3/4), \pi^*(1)) = 1/4$, and so on. This completes the definition of π^* over U.

With this construction, for any $r < r' \in U$, we have $\mathrm{d}(\pi^*(r), \pi^*(r')) \leqslant r' - r$. Thus, by Lemma 6.2(iii), we have $\|\pi^*(r)\pi^*(r')\| \leqslant r' - r$. In other words, π^* is Lipschitz over U. As U is dense in $[0,1]$, we can extend π^* to a Lipschitz (and thus continuous) function over $[0,1]$, by taking

$$\forall t \in [0,1], \pi^*(t) = \lim_{\substack{r \in U \\ r \to t}} \pi^*(r).$$

By Lemma 6.2(iv), we know that $\mathrm{d}(\cdot,\cdot)$ is continuous. Thus, as $\mathrm{d}(\pi^*(r), \pi^*(r')) \leqslant r' - r$ for any $r < r' \in U$, we conclude that $\mathrm{d}(\pi^*(t), \pi^*(t')) \leqslant t' - t$ for any $t < t' \in [0,1]$.

Finally, by definition, $C(\pi^*) = \sup_\sigma\{\sum_{i=1}^m \mathrm{d}(\pi^*(t_{i-1}), \pi^*(t_i))\}$ over all $[0,1]$-sequence $\sigma = (t_0, \ldots, t_m)$ such that $\pi_\sigma \subset |\mathcal{T}|$. We have proved that $\mathrm{d}(\pi^*(t_{i-1}), \pi^*(t_i)) \leqslant t_i - t_{i-1}$. So $\sum_{i=1}^m \mathrm{d}(\pi^*(t_{i-1}), \pi^*(t_i)) \leqslant 1$. Hence, $C(\pi^*) \leqslant 1 = \mathrm{d}(x,y)$. ◻

By the definition of a $\mathcal{T}$-rectifiable path, it follows that:

Corollary 4 *For any $\alpha > 1$ and for any $x, y \in |\mathcal{T}|$, there exists a polygonal path S^α from x to y such that $\text{cost}(S^\alpha) = C(S^\alpha) \leqslant \alpha\, C(\pi^*)$, where π^* is a shortest rectifiable path from x to y.*

Theorem 3 and Corollary 4 allow us to apply the results in Sections 3 and 5 to approximate the shortest rectifiable path. We summarize our main results in the following.

- Theorem 3 shows that there is a shortest rectifiable path. Furthermore, Corollary 4 shows that for any $\alpha > 1$, there exists a polygonal path with cost at most α times the optimal.

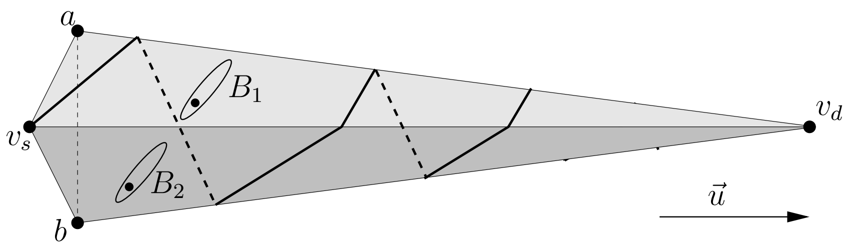

Figure 9: The terrain is formed by triangles abv_d, av_sv_d and bv_sv_d. The ellipses B_1 and B_2 are the unit balls of faces av_sv_d and bv_sv_d, respectively. The vertices a, b and v_s are fixed and the convex distance functions are fixed. When v_d goes to infinity in direction $\vec{u}$, the number of times a 1.01-approximate shortest path has to turn around the axis $(v_d, \vec{u})$ goes to infinity.

- Corollary 4 and Theorem 2 imply that for any $\varepsilon \in (0,1)$, there exists a $(21\rho n^2/\varepsilon)$-link path with cost at most $(1+\varepsilon)$ times the optimal.
- Corollary 4 and Theorem 2 allow us to apply Theorem 1 to approximate shortest paths in anisotropic regions. As stated in Corollary 2, for any $\varepsilon \in (0,1)$, we can compute in time $O\left(\frac{\rho^2 \log \rho}{\varepsilon^2} n^3 \log\left(\frac{\rho n}{\varepsilon}\right)\right)$ a polygonal path S with cost at most $(1+\varepsilon)$ times the optimal.

7 Conclusion

We have given algorithms for shortest paths problems in planar subdivisions. A natural question is whether our results can be generalized to higher dimensions. The algorithms for k-link paths that we presented in Section 3, as well as the proof of the existence of a shortest rectifiable path (Section 6), generalize directly to the case where $\mathcal{T}$ is a two-dimensional simplicial complex properly embedded in $\mathbb{R}^d$ (for any integer d).

However, our bound $O(\rho n^2/\varepsilon)$ on the number of links of an approximate shortest path (Theorem 2) does not generalize to higher dimensions. It does not even generalize to the case of a terrain. See the example shown in Figure 9, where $n = 4$, and ρ is fixed. The number of edges in a 1.01-approximate shortest path goes to infinity when v_d goes to infinity in direction $\vec{u}$.

References

[1] L. Aleksandrov, A. Maheshwari, and J.-R. Sack. Approximation algorithms for geometric shortest path problems. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 286–295, 2000.

[2] L. Aleksandrov, A. Maheshwari, and J.-R. Sack. Determining approximate shortest paths on weighted polyhedral surfaces. *Journal of the ACM*, 52(1):25–53, 2005.

[3] D. Burago, Y. Burago, and S. Ivanov. *A Course in Metric Geometry.* American Mathematical Society, 2001.

[4] P. Chew and R. Dyrsdale. Voronoi diagrams based on convex distance functions. In *Proceedings of the 1st Symposium on Computational Geometry*, pages 235–244, 1985.

[5] O. Daescu, J. Mitchell, S. Ntafos, J. Palmer, and C.-K. Yap. k-link shortest paths in weighted subdivisions. In *Proceedings of the 9th International Workshop on Algorithms and Data Structures*, pages 325–337, 2005.

[6] H. Edelsbrunner. *Geometry and Topology for Mesh Generation.* Cambridge University Press, 2001.

[7] S. Fortune. Voronoi diagrams and delaunay triangulations. In D.-Z. Du and F. Hwang, editors, *Computing in Euclidean Geometry.* World Scientific, 1995.

[8] M. Fredman and R. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615, 1987.

[9] H. Guggenheimer. *Differential Geometry.* Dover, 1977.

[10] U. Lang. Length spaces. Lecture notes, www.math.ethz.ch/˜lang/mg.pdf, 2006.

[11] M. Lanthier, A. Maheshwari, and J.-R. Sack. Shortest anisotropic paths on terrains. In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming*, pages 524–533, 1999.

[12] J. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 633–701. Elsevier, 2000.

[13] J. Mitchell and C. Papadimitriou. The weighted region problem: Finding shortest paths through a weighted planar subdivision. *Journal of the ACM*, 38(1):18–73, 1991.

[14] T. Nishizeki. *Planar Graphs : Theory and Algorithms.* Elsevier, 1988.

[15] F. Preparata and M. Shamos. *Computational Geometry: an Introduction.* Springer, 1985.

[16] J. Reif and Z. Sun. Movement planning in the presence of flows. *Algorithmica*, 39(2):127–153, 2004.

[17] J. Sellen. Direction weighted shortest path planning. In *Proceedings of the International Conference on Robotics and Automation*, pages 1970–1975, 1995.

[18] Z. Sun and J. Reif. On finding energy-minimizing paths on terrains. *IEEE Transactions on Robotics*, 21(1):102–114, 2005.

[19] Z. Sun and J. Reif. On finding approximate optimal paths in weighted regions. *Journal of Algorithms*, 58(1):1–32, 2006.

[20] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. Gortler, and H. Hoppe. Fast exact and approximate geodesics on meshes. *ACM Transactions on Graphics*, 24(3):553–560, 2005.