Querying Approximate Shortest Paths in Anisotropic Regions

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

December 3, 2006

Abstract

We present a data structure for answering approximate shortest path queries in a planar subdivision from a fixed source. Let $\rho \geqslant 1$ be a real number. Distances in each face of this subdivision are measured by a possibly asymmetric 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. Our data structure returns an $(1+\varepsilon)$ approximation of the shortest path cost from the fixed source to a query destination in $O(\log \frac{\rho n}{\varepsilon})$ time. Afterwards, an $(1+\varepsilon)$-approximate shortest path can be reported in time linear in its complexity. The data structure uses $O(\frac{\rho^2 n^4}{\varepsilon^2} \log \frac{\rho n}{\varepsilon})$ space and can be built in $O(\frac{\rho^2 n^4}{\varepsilon^2}(\log \frac{\rho n}{\varepsilon})^2)$ time. Our time and space bounds do not depend on any geometric parameter of the subdivision such as the minimum angle.

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

†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

1 Introduction

The problem of computing a shortest path between two points in a polygonal subdivision arises naturally in geographic information systems, VLSI design, logistics, and motion planning. We are interested in the case in which different metrics may be used in different faces of the subdivision, so as to model friction, wind, or any other constraint.

Efficient algorithms have been developed for finding $(1+\varepsilon)$-approximate shortest paths in some cases of non-Euclidean path costs. We mention several relevant results here. The readers are referred to a more detailed survey of various shortest path problems by Mitchell [10]. Let n be the size of a polygonal subdivision $\mathcal{T}$. In the weighted region problem [11], each face f of $\mathcal{T}$ is associated with a weight $w_f > 0$. The cost of a path within f is the length of this path multiplied by w_f. Mitchell and Papadimitriou [11] presented an $O(n^8L)$-time algorithm, where L is the maximum number of bits of the input numbers. Aleksandrov et al. [3] and Sun and Reif [13] gave algorithms that run in time linear in n. However, their time bounds depend on the weights and/or the minimum angle in $\mathcal{T}$. Reif and Sun considered a more general model that allows for uniform flows [12]. Cheng et al. [4] generalized the model further in which distances in a face is measured using a possibly asymmetric convex distance function, whose unit disk is contained in a concentric unit Euclidean disk, and contains a concentric Euclidean disk with radius $1/\rho$ for some $\rho \geqslant 1$. The running time of the algorithm by Cheng et al. depends on ρ, n, and ε, but not on the geometry of $\mathcal{T}$ such as the minimum angle.

Only the weighted region setting has been studied in answering approximate shortest path queries. Lanthier et al. [8] presented a data structure which can answer approximate shortest path queries for weighted polyhedral surfaces with additive error WL, where W is the maximum weight of the facets, and L is the length of the longest edge in the subdivision. Aleksandrov et al. [2] obtained some preliminary result for $(1+\varepsilon)$-approximate shortest path queries on weighted polyhedral surfaces. Recently, Aleksandrov et al. [1] presented improved results on querying $(1+\varepsilon)$-approximate shortest paths on weighted polyhedral surfaces of arbitrary genus. They allow both the source and destination to be specified in the query. In [1, 2], both the preprocessing time and the space depend on the weights and the minimum angle in the polyhedral surface.

We present a data structure to answer approximate shortest path queries from a fixed source in a polygonal subdivision. We adopt the model in [4] in which distances in faces are measured using possibly asymmetric convex distance functions. Our data structure reports an $(1+\varepsilon)$ approximation of the shortest path cost to a query destination in $O(\log \frac{\rho n}{\varepsilon})$ time. Afterwards, an $(1+\varepsilon)$-approximate shortest path can be output in time linear in its complexity. The data structure can be constructed in $O(\frac{\rho^2 n^4}{\varepsilon^2}(\log \frac{\rho n}{\varepsilon})^2)$ time and it uses $O(\frac{\rho^2 n^4}{\varepsilon^2} \log \frac{\rho n}{\varepsilon})$ space. Our time and space bounds do not depend on any geometric parameter of $\mathcal{T}$ such as the minimum angle.

Our result uses a variant of the discretization of $\mathcal{T}$ and the algorithm proposed in [4]. There are two difficulties to overcome in the query setting. First, it is unclear how to efficiently obtain a good lower bound on the shortest path cost when the query destination is very close to the fixed source. So it is difficult to establish the approximation bound. Second, consider the circles centered at the fixed source and passing through the other subdivision vertices. Suppose that two consecutive circles define an annulus of huge width. The query destination may fall anywhere in the annulus and a straightforward discretization of this annulus will introduce a spread-like geometric parameter into the preprocessing time and space. We introduce a scaling transformation and a subdivision perturbation technique. They map the query destinations in the above difficult cases to certain canonical positions. We can thus preprocess for the canonical positions to answer the queries.

2 Background

Environment. W.l.o.g., we assume that the faces of $\mathcal{T}$ are triangles. The *underlying space* $|\mathcal{T}| \subset \mathbb{R}^2$ is the union of all faces. We assume that $|\mathcal{T}|$ is connected, but we allow $|\mathcal{T}|$ to have holes so as to model obstacles. We assume that each *vertex* (resp. each *edge*) of $\mathcal{T}$ is a vertex (resp. an edge) of some face of $\mathcal{T}$. We do not allow dangling edges or isolated vertices. For any two points $p, q \in \mathbb{R}^2$, we denote by $\overline{pq}$ the closed, oriented line segment from p to q. 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. We use $\mathrm{int}(\cdot)$ and $\mathrm{bd}(\cdot)$ to denote interior and boundary, respectively, according to the usual topology of $\mathbb{R}^2$.

Convex Distance Functions. Each face f is associated with a compact convex set B_f that contains the origin. The convex distance function d_f is defined by $d_f(x, y) = \min\{\lambda \in [0, +\infty) : y \in x + \lambda B_f\}$. Since d_f may be asymmetric, it may not be a metric. Still, d_f satisfies the triangle inequality and the shortest path from p to q in f 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 $d_f(p, q)$. 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, 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 belongs to the interval $[1/\rho, 1]$. It implies that $\|pq\| \leqslant \mathrm{cost}(\overline{pq}) \leqslant \rho\,\|pq\|$.

Polygonal Paths. We use s to denote the fixed source and t to denote the query destination. W.l.o.g., we assume that s is a vertex of $\mathcal{T}$, which can be enforced by splitting triangles containing s if necessary. The query point t can be anywhere in $|\mathcal{T}|$. A *polygonal path* is a polyline in $|\mathcal{T}|$. 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 $\mathcal{T}$. If a polygonal path P has the node sequence $(p_0, p_1, \ldots, p_m)$, we write $P = (p_0, p_1, \ldots, p_m)$. The *length* of P is defined as $\mathrm{length}(P) = \sum_{i=1}^{m} \|p_{i-1}p_i\|$. For all integer k, a *k-link path* is a polygonal path such that: (i) there are at most k links, (ii) the link incident to the destination is contained in a face, (iii) the other links are either chords of faces or segments on face boundaries.*

Approximation. Cheng et al. [4] showed that the shortest path needs not be polygonal. They also proved the following results that we make use of later. We use $B(x, r)$ to denote a closed Euclidean disk centered at x with radius r.

Lemma 2.1 ([4])

(i) *There exists an $(1 + \varepsilon/6)$-approximate shortest polygonal path from s to t with at most k_ε links, where $k_\varepsilon = O(\rho n^2/\varepsilon)$.*

(ii) *If P is an $(1 + \varepsilon/3)$-approximate shortest polygonal path from s to t, then* $\mathrm{cost}(P) \leqslant \frac{4}{3}\rho\|st\|_{\mathcal{T}}$ *and* $P \subset B(s, \frac{4\rho\|st\|_{\mathcal{T}}}{3})$.

*Our definition of k-link paths is slightly more general than that given in [4] because we allow the destination to be anywhere in $|\mathcal{T}|$. Nevertheless, the results in [4] still hold by conceptually inserting the destination as a vertex of the subdivision.

3 A Building Block

Let r and R be two real numbers such that $r \leqslant \frac{3R}{4\rho}$. We describe a data structure that answers approximate shortest path queries for destinations t such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{3R}{4\rho}$. This data structure is a building block for our final result. It is based on a variant of the discretization technique in [4].

3.1 Data Structure

First, we discretize the edges of $\mathcal{T}$. Define $D_i = B(s, 2^i r)$ for $0 \leqslant i \leqslant \lceil \log \frac{R}{r} \rceil$, and define $D_{-1} = \emptyset$. For each edge e of $\mathcal{T}$ and for every integer $0 \leqslant i \leqslant \lceil \log \frac{R}{r} \rceil$, we insert the intersection points between e and the boundary of D_i and then place a maximal set of Steiner points on $\text{int}(e \cap (D_i \setminus D_{i-1}))$ with uniform spacing $\delta_i = \frac{\epsilon}{24\rho k_\varepsilon} 2^i r$. (The term k_ε is from Lemma 2.1(i).) Define $V(r, R, \varepsilon)$ to be the set of Steiner points created above and the vertices of $\mathcal{T}$ in $D_{\lceil \log R/r \rceil}$. Clearly, $|V(r, R, \varepsilon)| = O(\frac{\rho n k_\varepsilon}{\epsilon} \log \frac{R}{r}) = O(\frac{\rho^2 n^3}{\epsilon^2} \log \frac{R}{r})$ by Lemma 2.1(i).

For any two points $p_i, p_j \in V(r, R, \varepsilon)$ that lie on $\text{bd}(f)$ for some face f, imagine that we introduce a directed edge (p_i, p_j) with weight equal to $\text{cost}(\overline{p_i p_j})$. This produces a weighted directed graph $G(r, R, \varepsilon)$. We emphasize that $G(r, R, \varepsilon)$ is defined for notational convenience and for the analysis. It is never computed.

For any $p_i \in V(r, R, \varepsilon)$, we precompute the shortest path in $G(r, R, \varepsilon)$ from s to p_i. (We describe how this can be done without computing $G(r, R, \varepsilon)$ in the proof of Lemma 3.2.) Define the weight $w(p_i)$ of p_i as the cost of this path. One can show that for any $t \in f$, $\min\{\, w(p_i) + d_f(p_i, t) : p_i \in V(r, R, \varepsilon) \cap \text{bd}(f) \,\}$ is an $(1+\varepsilon)$ approximation of the shortest path cost to t. This inspires the construction of an additively weighted Voronoi diagram of points in $V(r, R, \varepsilon) \cap \text{bd}(f)$ using the convex distance function d_f. This gives us the desired performance (as stated in Lemma 3.2); however, since d_f is arbitrary, one has to assume the ability to compute and intersect bisectors in $O(1)$ time, and to process the Voronoi diagram for point location.

To eliminate these assumptions, we propose to approximate the unit ball B_f by a convex polygon $\widehat{B}_f$. This defines a distance function $\widehat{d}_f$ which approximates d_f. Recall that B_f contains the origin o. We take $\widehat{B}_f$ to be the convex hull of $\Theta(\frac{\rho}{\varepsilon})$ carefully placed points on $\text{bd}(B_f)^\dagger$ such that for any ray γ emitting from o, $\text{length}(\gamma \cap \widehat{B}_f) \leqslant \text{length}(\gamma \cap B_f) \leqslant (1 + \frac{\varepsilon}{10})\text{length}(\gamma \cap \widehat{B}_f)$. That is, $\frac{1}{1+\varepsilon/10} \widehat{d}_f(x, y) \leqslant d_f(x, y) \leqslant \widehat{d}_f(x, y)$. By breaking ties in a "lexicographical manner" [7], Ma [9] showed that if the distance function is defined by a convex polygon of K sides, then the Voronoi regions are star-shaped and every bisector is an open polygonal line of $O(K)$ segments. We can extend these results to the additively weighted case. Thus, a bisector has $O(\frac{\rho}{\varepsilon})$ segments in our case and it can be computed in $O(\frac{\rho}{\varepsilon})$ time. We use the divide-and-conquer algorithm of Chew and Drysdale [5] to construct the Voronoi diagram.

To obtain the desired performance we cannot afford to compute the Voronoi diagram of $V(r, R, \varepsilon) \cap \text{bd}(f)$. It is in fact unnecessary. For every edge e of f and for each integer $0 \leqslant i \leqslant \lceil \log \frac{R}{r} \rceil$, we put every k_ε consecutive Steiner points in $e \cap (D_i \setminus D_{i-1})$ in a cluster. In each cluster we designate the Steiner point p_j whose weight $w(p_j)$ is minimum as the *cluster hub*. The Voronoi diagram of the cluster hubs in $\text{bd}(f)$ already serves our purposes. Nevertheless, due to a technical requirement of the divide-and-conquer approach, we need to construct one Voronoi diagram for the cluster hubs on each edge of f. The "closest" cluster hub can be found by performing a point location in the Voronoi diagram for each edge of f and then taking the

†Given a direction, one can pick any point p in that direction, compute $d_f(o, p)$, and scale the distance to obtain a point in $\text{bd}(B_f)$.

minimum. There is no loss in asymptotic efficiency because f is a triangle.

Given a point set S, the divide-and-conquer algorithm takes $O((T_1 + T_2)|S|\log|S|)$ time, where T_1 is the time to traverse a bisector and T_2 is the time to intersect two bisectors. In our case $T_1 = O(\frac{\rho}{\varepsilon})$. Whenever two bisectors are intersected in the algorithm, they are defined by two point-pairs that share a common point. Since the regions bounded by both bisectors are star-shaped with respect to this common point, there are $O(\frac{\rho}{\varepsilon})$ intersections which can be computed by simply traversing the bisectors in $O(\frac{\rho}{\varepsilon})$ time. Hence, the Voronoi diagram can be constructed in $O(\frac{\rho}{\varepsilon}|S|\log|S|)$ time.

Given a query destination t such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{3R}{4\rho}$, we use a point location structure for $\mathcal{T}$ to find the face f containing t. Then, we perform point locations in the Voronoi diagrams associated with f. This gives us the cluster hub that achieves $\min\{\, w(p_i) + \widehat{d}_f(p_i,t) : p_i \in \mathrm{bd}(f) \text{ is a cluster hub}\,\}$. Denote this cluster hub by h_t. We report $w(h_t) + d_f(h_t,t)$ as the approximate path cost. The approximate shortest path is the precomputed shortest path in $G(r,R,\varepsilon)$ from s to h_t followed by $\overline{h_t t}$.

3.2 Analysis

Lemma 3.1 *Assume that $h_t \in \mathrm{bd}(f)$. Let P be the shortest path in $G(r,R,\varepsilon)$ from s to h_t. Then $\mathrm{cost}(P) + d_f(h_t,t) \leqslant (1+\varepsilon)\mathsf{OPT}$, where OPT is the shortest path cost from s to t.*

Proof. By Lemma 2.1(i), there exists a k_ε-link path Q from s to t whose cost is within $(1+\varepsilon/6)$ of the optimal. By lemma 2.1(ii) and our assumption that $\|st\|_{\mathcal{T}} \leqslant \frac{3R}{4\rho}$, $Q \subset B(s, \frac{4\rho\|st\|_{\mathcal{T}}}{3}) \subset B(s,R)$. Since Q is a k_ε-link path, any node of Q other than s and t lies on some edge e of $\mathcal{T}$. Let $x \in \mathrm{bd}(f)$ be the last entry point of Q into f. Note that x is a node of Q.

We snap every node w of Q other than s, x, and t as follows. Assume that $w \in D_i \setminus D_{i-1}$ for some $0 \leqslant i \leqslant \lceil \log \frac{R}{r} \rceil$. By construction, there is a Steiner point on e at distance $\frac{\varepsilon}{24\rho k_\varepsilon}2^i r$ or less from w. We snap w to this Steiner point and the path cost increases by at most $\frac{\varepsilon}{12k_\varepsilon}2^i r$. Observe that $\mathrm{cost}(Q) \geqslant \|sw\|_{\mathcal{T}} \geqslant 2^{i-1}r$ and so the additional cost is bounded by $\frac{\varepsilon}{6k_\varepsilon}\mathrm{cost}(Q)$. Since Q has at most k_ε nodes, the total extra cost is at most $\frac{\varepsilon}{6}\mathrm{cost}(Q) \leqslant \frac{\varepsilon}{6}(1+\frac{\varepsilon}{6})\mathsf{OPT} < \frac{\varepsilon}{3}\mathsf{OPT}$.

We snap x to the nearest cluster hub p_j, whose distance from x is at most $\frac{\epsilon}{24\rho}2^i r \leqslant \frac{\varepsilon}{12\rho}\mathrm{cost}(Q) < \frac{\varepsilon}{6\rho}\mathsf{OPT}$. So the cost of this detour is at most $\frac{\varepsilon}{3}\mathsf{OPT}$.

In all, the cost of the modified path from s to t, which goes through the cluster hub p_j, is at most $\mathrm{cost}(Q) + \frac{\varepsilon}{3}\mathsf{OPT} + \frac{\varepsilon}{3}\mathsf{OPT} \leqslant (1+\frac{5\varepsilon}{6})\mathsf{OPT}$. Observe that this modified path is the concatenation of a path in $G(r,R,\varepsilon)$ from s to p_j and $\overline{p_j t}$. We have $w(h_t) + d_f(h_t,t) \leqslant w(h_t) + \widehat{d}_f(h_t,t) \leqslant w(p_j) + \widehat{d}_f(p_j,t) \leqslant (1+\frac{\varepsilon}{10})(w(p_j) + d_f(p_j,t)) \leqslant (1+\frac{\varepsilon}{10})(1+\frac{5\varepsilon}{6})\mathsf{OPT} < (1+\varepsilon)\mathsf{OPT}$ for small ε. ◻

The following lemma summarizes the performance of the data structure.

Lemma 3.2 *Let r and R be two numbers such that $r \leqslant \frac{3R}{4\rho}$. There is a data structure* $\mathsf{PathQuery}(r,R,\varepsilon)$ *such that for any $t \in |\mathcal{T}|$ where $r \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{3R}{4\rho}$, the cost of an $(1+\varepsilon)$-approximate shortest path from s to t can be reported in $O(\log\frac{\rho n}{\varepsilon} + \log\log\frac{R}{r})$ time. The approximate path can be output in time linear in its complexity, which is $O(\frac{\rho^2 n^3}{\varepsilon^2}\log\frac{R}{r})$.* $\mathsf{PathQuery}(r,R,\varepsilon)$ *uses $O(\frac{\rho^2 n^3}{\varepsilon^2}\log\frac{R}{r})$ space and can be built in $O((\frac{\rho^2 n^3}{\varepsilon^2}\log\frac{R}{r})(\log\frac{\rho n}{\varepsilon} + \log\log\frac{R}{r}))$ time.*

Proof. The correctness has been proved in Lemma 3.1. Querying involves a point location in $\mathcal{T}$ and point locations in the Voronoi diagrams of the cluster hubs in a face of $\mathcal{T}$. The first query takes $O(\log n)$ time. Since the number of cluster hubs in a face is $O(\frac{\rho}{\varepsilon}\log\frac{R}{r})$, the second query takes $O(\log\frac{\rho}{\varepsilon} + \log\log\frac{R}{r})$ time. The path complexity follows from the bound on $|V(r,R,\varepsilon)|$.

To precompute the shortest paths in $G(r, R, \varepsilon)$ to points in $V(r, R, \varepsilon)$, we run the BUSHWHACK algorithm [13] as in [4]. The BUSHWHACK algorithm discovers a shortest path tree in $G(r, R, \varepsilon)$ from s to all other vertices. The advantage of the BUSHWHACK algorithm is that the edges in $G(r, R, \varepsilon)$ need not be generated and the algorithm runs in $O(nm \log nm)$ time, where m is the number of Steiner points placed on each edge. Since $m = O(\frac{\rho^2 n^2}{\varepsilon^2} \log \frac{R}{r})$ in our case, the running time is thus $O((\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{R}{r})(\log \frac{\rho n}{\varepsilon} + \log\log \frac{R}{r}))$. Storing the shortest path tree takes $O(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{R}{r})$ space.

The point location structure for $\mathcal{T}$ takes $O(n)$ space and can be built in $O(n \log n)$ time. The construction of the Voronoi diagrams for a face takes $O(\frac{\rho}{\varepsilon} h \log h)$ time, where h is the number of cluster hubs in a face. Since $h = O(\frac{\rho}{\varepsilon} \log \frac{R}{r})$ and there are $O(n)$ faces, the total time is $O((\frac{\rho^2 n}{\varepsilon^2} \log \frac{R}{r})(\log \frac{\rho}{\varepsilon} + \log\log \frac{R}{r}))$. Constructing the point location structures for these Voronoi diagrams take the same time and the space is $O(\frac{\rho}{\varepsilon} hn) = O(\frac{\rho^2 n}{\varepsilon^2} \log \frac{R}{r})$. ◻

4 Close Range Queries

The data structure in Lemma 3.2 cannot handle query destinations that are arbitrarily close to s because $\frac{R}{r}$ becomes infinite if we set r to zero. The structure of the shortest path may be non-trivial even if the query destination lies within the union of faces of $\mathcal{T}$ incident to s. Figure 1 is an example from [4] such that the shortest path from s to t is not a polygonal path. It can be shown that the path intersects some edge incident to s an infinite number of times.

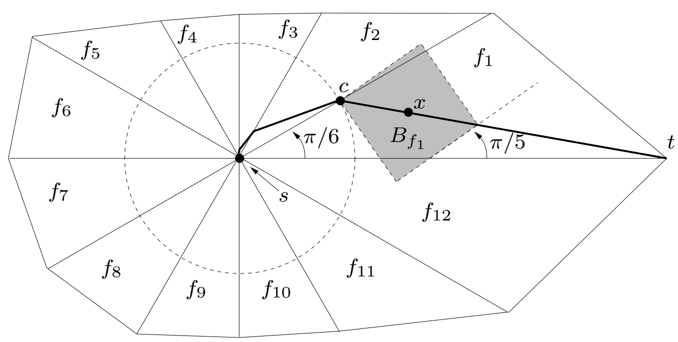

Figure 1: B_{f_1} is a square centered at the origin and with edge length $\sqrt{2}$. B_{f_i} is obtained by rotating B_{f_1} by an angle $(i-1)\pi/6$. We conjecture that the shortest path consists of an infinite sequence of segments spiraling around s.

Let ℓ_s be the distance from s to the nearest non-incident edge in $\mathcal{T}$. We propose a scaling transformation to handle query destinations that fall inside $B(s, \frac{\ell_s}{2\rho})$. First, by Lemma 2.1, for any $t \in B(s, \frac{\ell_s}{2\rho}) \cap |\mathcal{T}|$, there is an $(1+\varepsilon)$-approximate shortest polygonal path that lies inside $B(s, \ell_s)$. It is easier to work with $B(s, \frac{\ell_s}{2\rho}) \cap |\mathcal{T}|$ as $B(s, \ell_s)$ intersects faces incident to s only. Let t' be the intersection point between the boundary of $B(s, \frac{\ell_s}{2\rho})$ and the ray emitting from s through t. We define the scaling transformation $\phi : \mathbb{R}^2 \to \mathbb{R}^2$ as follows.

Definition 1 *For any $p \in \mathbb{R}^2$, define $\phi(p) = s + \frac{\|st'\|}{\|st\|}(p - s)$.*

Note that $\phi(t) = t'$ and $\phi(s) = s$. Clearly, ϕ^{-1} is well defined. We can apply ϕ to a polygonal path P from s to t by applying ϕ to the nodes of P. The cost of the path is scaled properly as stated in the following result. Note that both $B(s, 2\rho\|st\|)$ and $B(s, 2\rho\|st'\|)$ intersect the faces incident to s only.

Lemma 4.1 *Let $P = (s = p_0, p_1, \ldots, p_m = t)$ be a polygonal path. Let $\phi(P) = (s = \phi(p_0), \phi(p_1), \ldots, \phi(p_m) = t')$.*

(i) $P \subset B(s, 2\rho\|st\|) \cap |\mathcal{T}| \iff \phi(P) \subset B(s, 2\rho\|st'\|) \cap |\mathcal{T}|$.

(ii) *If $P \subset B(s, 2\rho\|st\|) \cap |\mathcal{T}|$ or $\phi(P) \subset B(s, 2\rho\|st'\|) \cap |\mathcal{T}|$, then $\mathrm{cost}(\phi(P)) = \frac{\|st'\|}{\|st\|}\mathrm{cost}(P)$.*

Lemma 4.1 allows us to show that approximate shortest paths are preserved under the scaling transformation. Notice that $\|st\| = \|st\|_{\mathcal{T}}$ and $\|st'\| = \|st'\|_{\mathcal{T}}$ because s, t and t' are contained in the same face incident to s.

Lemma 4.2 *If a polygonal path Q is an $(1+\varepsilon)$-approximate shortest path from s to t' in $B(s, 2\rho\|st'\|) \cap |\mathcal{T}|$, then $\phi^{-1}(Q)$ is an $(1+\varepsilon)$-approximate shortest path from s to t.*

Proof. Let $\mathcal{P}(x)$ denote the collection of all polygonal paths in $\mathcal{T}$ from s to a point x.

$$\begin{aligned}
\mathrm{cost}(\phi^{-1}(Q)) &= \frac{\|st\|}{\|st'\|}\mathrm{cost}(Q) \\
&\leqslant \frac{\|st\|}{\|st'\|}(1+\varepsilon)\inf\left\{\mathrm{cost}(R') : R' \in \mathcal{P}(t')\right\} \\
&\leqslant \frac{\|st\|}{\|st'\|}(1+\varepsilon)\inf\left\{\mathrm{cost}(R') : R' \in \mathcal{P}(t') \text{ and } R' \subset B(s, 2\rho\|st'\|)\right\} \\
&= (1+\varepsilon)\inf\left\{\mathrm{cost}(R) : R \in \mathcal{P}(t) \text{ and } R \subset B(s, 2\rho\|st\|)\ \right\}.
\end{aligned}$$

The cost of any path from s to t that leaves $B(s, 2\rho\|st\|)$ is more than the cost of $\overline{st}$ which lies inside $B(s, 2\rho\|st\|)$. So we conclude that $\mathrm{cost}(\phi^{-1}(Q)) \leqslant (1+\varepsilon)\inf\{\mathrm{cost}(R) : R \in \mathcal{P}(t)\}$. ◻

Lemma 4.2 allows us to make an equivalent query for t' on the boundary of $B(s, \frac{\ell_s}{2\rho})$. Therefore, we can use the data structure $\mathsf{PathQuery}(\frac{\ell_s}{2\rho}, \frac{2\ell_s}{3}, \varepsilon)$ in Lemma 3.2, which yields the following result.

Lemma 4.3 *There is a data structure* $\mathsf{CloseRange}(\varepsilon)$ *such that for any $t \in B(s, \frac{\ell_s}{2\rho}) \cap |\mathcal{T}|$, the cost of an $(1+\varepsilon)$-approximate shortest path from s to t can be reported in $O(\log\frac{\rho n}{\varepsilon})$ time. The approximate path can be output in time linear in its complexity, which is $O(\frac{\rho^2 n^3}{\varepsilon^2}\log\rho)$.* $\mathsf{CloseRange}(\varepsilon)$ *uses $O(\frac{\rho^2 n^3}{\varepsilon^2}\log\rho)$ space and can be built in $O((\frac{\rho^2 n^3}{\varepsilon^2}\log\rho)\log\frac{\rho n}{\varepsilon})$ time.*

5 The Complete Data Structure

For technical purposes, we split some edges of $\mathcal{T}$ as follows. For every edge e of $\mathcal{T}$, if some ball centered at s makes a tangential contact with e at an interior point q, then we split e at q into two edges. There are still $O(1)$ edges in the boundary of every face. Notice that this processing does not affect Lemma 3.2 and Lemma 4.3, and the size of the processed subdivision is within a constant factor of the original size. For convenience, we still use $\mathcal{T}$ to denote the processed subdivision and we still use n to denote the number of vertices afterwards. Now, for any edge e, no ball centered at s makes a tangential contact with e at any interior point.

Let $s = v_0, v_1, v_2, v_3, \ldots$ be the vertices of $\mathcal{T}$ sorted in non-decreasing geodesic distances from s. For each vertex v_i, we define

$$r_i = \frac{1}{2\rho}\|sv_i\|_{\mathcal{T}}, \qquad R_i = \frac{c}{\varepsilon^3}\rho^4 n^3(\log\rho)\|sv_i\|_{\mathcal{T}},$$

where $c \geqslant 4$ is a positive constant. We will explain the setting of c later. Notice that $r_1 = \frac{\ell_s}{2\rho}$. We compute the data structures $\mathsf{CloseRange}(\varepsilon)$ and $\mathsf{PathQuery}(r_i, R_i, \varepsilon)$ for all v_i. Given a query destination t, we first obtain $\|st\|_{\mathcal{T}}$ using the data structure in [6].

If $\|st\|_{\mathcal{T}} < \frac{\ell_s}{2\rho} = r_1$, we use $\mathsf{CloseRange}(\varepsilon)$ to answer the query in $O(\log\frac{\rho n}{\varepsilon})$ time. Otherwise, we locate the pair of vertices v_a and v_{a+1} such that $r_a \leqslant \|st\|_{\mathcal{T}} < r_{a+1}$. If $r_a \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{3R_a}{4\rho}$, we use $\mathsf{PathQuery}(r_a, R_a, \varepsilon)$ to answer the query in $O(\log\frac{\rho n}{\varepsilon})$ time. The remaining case is that $\frac{3R_a}{4\rho} < \|st\|_{\mathcal{T}} < r_{a+1}$. We describe a data structure to answer such a query in $O(\log\frac{\rho n}{\varepsilon})$ time. It is based on a local perturbation of the subdivision followed by applying the scaling transformation. In the rest of this section a denotes the index such that $\frac{3R_a}{4\rho} < \|st\|_{\mathcal{T}} < r_{a+1}$.

5.1 Structural Properties

Define an annulus:

$$A(v_a) = B\left(s, \frac{4\rho r_{a+1}}{3}\right) \setminus B(s, 2\rho r_a).$$

Note that $\frac{4\rho r_{a+1}}{3} = \frac{2}{3}\|sv_{a+1}\|_{\mathcal{T}}$ and $2\rho r_a = \|sv_a\|_{\mathcal{T}}$. We prove several structural properties that are useful in developing the perturbation technique in the next subsection.

First, we show that if $\frac{3R_a}{4\rho} < \|st\|_{\mathcal{T}} < r_{a+1}$, then $t \in |A(v_a) \cap \mathcal{T}|$. It is clear that $t \in B(s, \frac{4\rho r_{a+1}}{3})$. Although $\frac{3R_a}{4\rho} > 2\rho r_a$, it is not immediately obvious why $t \notin B(s, 2\rho r_a)$ because the geodesic distance is not equal to the straight-line distance in the presence of obstacles. We need the following technical lemma.

Lemma 5.1 *Assume that $\frac{3R_a}{4\rho} < r_{a+1}$. For any vertex $v_i \in B(s, \frac{4\rho r_{a+1}}{3})$, if there is a polygonal path P from s to v_i inside $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$, then $\|sv_i\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$ and hence $v_i \in B(s, \|sv_a\|_{\mathcal{T}}) = B(s, 2\rho r_a)$.*

Proof. W.l.o.g., assume that P is a shortest path from s to v_i inside $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$, and thus all its nodes are vertices of $\mathcal{T}$. We prove by induction that $\|sv_k\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$ for every node v_k of P. The base case is trivial as s is the first node of P. Assume that $\|sv_j\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$ for some node v_j of P. Let v_k be the next node along P. Since $P \subset B(s, \frac{4\rho r_{a+1}}{3})$, we have $\|sv_k\| \leqslant \frac{4\rho r_{a+1}}{3} = \frac{2}{3}\|sv_{a+1}\|_{\mathcal{T}}$. Therefore, $\|sv_k\|_{\mathcal{T}} \leqslant \|sv_j\|_{\mathcal{T}} + \|v_jv_k\| \leqslant \|sv_j\|_{\mathcal{T}} + \|sv_j\| + \|sv_k\| \leqslant 2\|sv_j\|_{\mathcal{T}} + \frac{2}{3}\|sv_{a+1}\|_{\mathcal{T}}$. By the inductive assumption, $\|sv_j\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$ and so $\|sv_k\|_{\mathcal{T}} \leqslant 2\|sv_a\|_{\mathcal{T}} + \frac{2}{3}\|sv_{a+1}\|_{\mathcal{T}}$. By the definition of R_a, $2\|sv_a\|_{\mathcal{T}} \leqslant R_a/2$. Since $\frac{3R_a}{4\rho} < r_{a+1}$ by assumption, we get $2\|sv_a\|_{\mathcal{T}} < \frac{2\rho}{3}r_{a+1} = \frac{1}{3}\|sv_{a+1}\|_{\mathcal{T}}$. Hence, $\|sv_k\|_{\mathcal{T}} < \|sv_{a+1}\|_{\mathcal{T}}$. The ordering of vertices by their geodesic distances implies that $\|sv_k\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$. ◻

Lemma 5.2 *For any $t \in |\mathcal{T}|$, if $\frac{3R_a}{4\rho} < \|st\|_{\mathcal{T}} < r_{a+1}$, then $t \in |A(v_a) \cap \mathcal{T}|$.*

Proof. Since $\|st\|_{\mathcal{T}} < r_{a+1}$, it is clear that $t \in B(s, \frac{4\rho r_{a+1}}{3})$. Assume to the contrary that $t \in B(s, 2\rho r_a) = B(s, \|sv_a\|_{\mathcal{T}})$. Because $\|st\|_{\mathcal{T}} < r_{a+1}$, there is a geodesic path P from s to t that lies inside $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$. Let v_i be the node preceding t in P. So

$\|st\|_{\mathcal{T}} = \|sv_i\|_{\mathcal{T}} + \|v_it\|$. By Lemma 5.1, $\|sv_i\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$ and $v_i \in B(s, \|sv_a\|_{\mathcal{T}})$. So $\|st\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}} + \|v_it\|$. Since $t \in B(s, \|sv_a\|_{\mathcal{T}})$ by assumption, $\|v_it\| \leqslant 2\|sv_a\|_{\mathcal{T}}$, which implies that $\|st\|_{\mathcal{T}} \leqslant 3\|sv_a\|_{\mathcal{T}} = 6\rho r_a$ which is at most $\frac{3R_a}{4\rho}$ by the definition of R_a. This contradicts the condition that $\|st\|_{\mathcal{T}} > \frac{3R_a}{4\rho}$. ◻

If there are obstacles, $A(v_a)$ may contain some vertices and $|A(v_a) \cap \mathcal{T}|$ may be disconnected. $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$ may also be disconnected. Define $A(v_a)|_t$ to be the connected component of $A(v_a) \cap \mathcal{T}$ whose underlying space contains t. The next result shows that $A(v_a)|_t$ is empty of vertices of $\mathcal{T}$.

Lemma 5.3 *Let $t \in |\mathcal{T}|$ be a point such that $\frac{3R_a}{4\rho} < \|st\|_{\mathcal{T}} < r_{a+1}$.*

(i) *If $A(v_a)$ contains a vertex v_i of $\mathcal{T}$, s and v_i lie in different connected components in $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$.*

(ii) *$A(v_a)|_t$ does not contain any vertex of $\mathcal{T}$.*

Proof. Since $A(v_a)$ contains v_i, $v_i \notin B(s, 2\rho r_a) = B(s, \|sv_a\|_{\mathcal{T}})$. The contrapositive of Lemma 5.1 implies that any polygonal path from s to v_i must leave $B(s, \frac{4\rho r_{a+1}}{3})$. That is, s and v_i lie in different connected components in $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$. This proves (i). By Lemma 2.1, $\|st\|_{\mathcal{T}} < r_{a+1}$ implies the existence of an $(1 + \varepsilon)$-approximate shortest path P in $B(s, \frac{4\rho\|st\|_{\mathcal{T}}}{3}) \subset B(s, \frac{4\rho r_{a+1}}{3})$. So s and t lie in the same connected component C in $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$. Observe that $A(v_a)|_t \subseteq C \setminus B(s, 2\rho r_a)$. Then, (ii) follows from (i). ◻

5.2 Perturbation

We look for an approximate shortest path within $A(v_a)|_t \cup B(s, 2\rho r_a)$. We will see later that restricting our attention to $A(v_a)|_t \cup B(s, 2\rho r_a)$ worsens the approximation bound negligibly.

Let $E_t = \{e_1, e_2, \ldots\}$ be the set of edges of $\mathcal{T}$ that intersect $A(v_a)|_t$. By Lemma 5.3, $A(v_a)|_t$ does not contain any endpoint of e_i. Moreover, due to our preprocessing, no ball centered at s makes a tangential contact with any interior point of e_i. So one endpoint of e_i lies outside $B(s, \frac{4\rho r_{a+1}}{3})$ and the other endpoint lies inside $B(s, 2\rho r_a)$. This property allows us to order the edges in E_t in clockwise order around $A(v_a)|_t$. Without loss of generality, we assume that e_{i+1} follows e_i in the clockwise order. Refer to Figure 2 for an example.

We want to map t to certain canonical positions so as to allow for preprocessing. The key insight is that if we ignore the parts of $\mathcal{T}$ inside $B(s, 2\rho r_a)$, the picture is very similar to the situation in close range queries. So there is hope for applying the scaling transformation here. The difference is that the edges e_i are not incident to s and the support lines of e_i's do not pass through s. Therefore, we perturb the e_i's so that their support lines pass through s. It allows us to apply the scaling transformation. We have to argue that the perturbations of the e_i's introduce relatively little error. We have ignored the parts of $\mathcal{T}$ inside $B(s, 2\rho r_a)$ so far. We argue that this introduces relatively little error too.

Let u_i be the intersection point between e_i and the boundary of $B(s, \frac{4\rho r_{a+1}}{3})$. Define the ray ℓ_i from s through u_i, which can be viewed as a perturbation of the support line of e_i. See Figure 2 for an example. The rays ℓ_i's partition $\mathbb{R}^2$ into cones. If e_i and e_{i+1} bound a face f of $\mathcal{T}$, then the cone bounded by ℓ_i and ℓ_{i+1} inherits the distance function d_f. Otherwise, the cone bounded by ℓ_i and ℓ_{i+1} is an obstacle cone. In all, the cones associated with distance functions form a subdivision. This subdivision has only one vertex s. All triangles are unbounded and

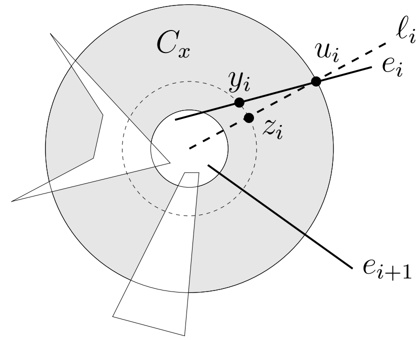

Figure 2: The white polygons are obstacles. The largest shaded region is $A(v_a)|_t$.

they are incident to s. We denote this subdivision by $T(v_a)$. $A(v_a) \cap T(v_a)$ can be viewed as a deformed version of $A(v_a)|_t$.

The above perturbation can be extended to the entire $A(v_a)|_t$ as follows. We need some notation first. For any point $x \in A(v_a)|_t$, let C_x be the circle that is centered at s and passes through x. Assume that x lies between edges e_i and e_{i+1}. Let $y_i = C_x \cap e_i$ and $y_{i+1} = C_x \cap e_{i+1}$. Let $z_i = C_x \cap \ell_i$ and $z_{i+1} = C_x \cap \ell_{i+1}$. Given two points $\alpha, \beta \in C_x$, we use $C_x[\alpha, \beta]$ to denote the arc on C_x from α to β in clockwise order. We can now define the perturbation φ from the underlying space of $A(v_a)|_t$ to that of $A(v_a) \cap T(v_a)$.

Definition 2 *For any point x in the underlying space of $A(v_a)|_t$, $\varphi(x)$ is the point on $C_x[z_i, z_{i+1}]$ such that*

$$\frac{\text{length}(C_x[z_i, \varphi(x)])}{\text{length}(C_x[z_i, z_{i+1}])} = \frac{\text{length}(C_x[y_i, x])}{\text{length}(C_x[y_i, y_{i+1}])}.$$

Clearly, φ is a bijection. The next result bounds the distance between x and $\varphi(x)$.

Lemma 5.4 *For any point $x \in A(v_a)|_t$, $\|\varphi(x)\, x\| \leqslant \pi \|sv_a\|_{\mathcal{T}}/2$.*

Proof. (Sketch) It suffices to show that the length of the arc on C_x between x and $\varphi(x)$ is at most $\pi\|sv_a\|_{\mathcal{T}}/2$. Assume that x lies on the arc $C_x[y_i, y_{i+1}]$. The length of $C_x([z_i, \varphi(x)])$ is linear in the length of $C_x[y_i, x]$. Therefore, the length of the arc between x and $\varphi(x)$ is maximized when $x = y_i$ or y_{i+1}. That is, it suffices to bound the length of the arc between y_i and z_i and the arc between y_{i+1} and z_{i+1}. Observe that both e_i and ℓ_i have endpoints inside $B(s, 2\rho r_a)$. Then, using elementary trigonometry, one can prove a bound of $\pi \rho r_a = \pi \|sv_a\|_{\mathcal{T}}/2$. ◻

We build the data structure $\mathsf{CloseRange}(\varepsilon/6)$ in Lemma 4.3 for $T(v_a)$ to answer an $(1+\varepsilon/6)$-approximate shortest path query from s to $\varphi(t)$. (In order to invoke this data structure, we set $\ell_s = \frac{8\rho^2 r_{a+1}}{3}$ to ensure that there is an approximate path from s to $\varphi(t)$ in $B(s, \frac{\ell_s}{2\rho}) \cap |T(v_a)| = B(s, \frac{4\rho r_{a+1}}{3}) \cap |T(v_a)|$.) Let P be the $(1+\varepsilon/6)$-approximate shortest path obtained. Let m be the number of nodes on P.[‡] We claim that $\text{cost}(P) + (2n + m\pi)\rho\|sv_a\|_{\mathcal{T}}$ is an $(1+\varepsilon)$-approximation of the shortest path cost from s to t. We also show how to transform P to a path X in $|\mathcal{T}|$ such that $\text{cost}(X) \leqslant \text{cost}(P) + (2n + m\pi)\rho\|sv_a\|_{\mathcal{T}}$, i.e., X is an $(1+\varepsilon)$-approximate shortest path.

We describe the transformation from P to X. Let b be the point at which P crosses the boundary of $B(s, 2\rho r_a)$ for the last time, and so into the $|A(v_a) \cap T(v_a)|$. We insert b as a new

[‡]Recall that $\text{cost}(P)$ is reported first. It is easy to extend the data structure to report the number of nodes on P in $O(1)$ time after reporting $\text{cost}(P)$.

node into P. Let the resulting sequence of nodes in P be $(s = p_0, p_1, \ldots, p_l = b, \ldots, p_m = \varphi(t))$. Compute the geodesic path X' from s to $\varphi^{-1}(b)$ in $B(s, 2\rho r_a) \cap |\mathcal{T}|$. Define the path $X'' = (\varphi^{-1}(b) = \varphi^{-1}(p_l), \varphi^{-1}(p_{l+1}), \ldots, \varphi^{-1}(p_m) = t)$. Then we define X to be the concatenation of X' and X''.

The number of segments in X' is at most n, each of length at most $4\rho r_a$. So $\mathrm{cost}(X') \leqslant 4n\rho^2 r_a = 2n\rho\|sv_a\|_{\mathcal{T}}$. By the triangle inequality, $\mathrm{cost}(X'') \leqslant \mathrm{cost}(P) + \sum_{i=l}^{m} 2\rho\|\varphi(p_i)\, p_i\|$, which is at most $\mathrm{cost}(P) + m\pi\rho\|sv_a\|_{\mathcal{T}}$ by Lemma 5.4. Hence, it is correct to report $\mathrm{cost}(P) + (2n + m\pi)\rho\|sv_a\|_{\mathcal{T}}$ as an upper bound of $\mathrm{cost}(X)$.

Next, we show that $\mathrm{cost}(P) + (2n + m\pi)\rho\|sv_a\|_{\mathcal{T}}$ is at most $(1 + \varepsilon)\mathsf{OPT}$, where OPT is the shortest path cost from s to t in $|\mathcal{T}|$. By Lemma 4.3, $m = O(\frac{\rho^2 n^3}{\varepsilon^2} \log \rho)$. So $(2n+m\pi)\rho\|sv_a\|_{\mathcal{T}} = O((\frac{\rho^3 n^3}{\varepsilon^2} \log \rho)\|sv_a\|_{\mathcal{T}})$. We choose c in the definition of R_a so that $(2n+m\pi)\rho\|sv_a\|_{\mathcal{T}} \leqslant \frac{\varepsilon}{2} \cdot \frac{3R_a}{4\rho} < \frac{\varepsilon}{2}\|st\|_{\mathcal{T}} \leqslant \frac{\varepsilon}{2}\mathsf{OPT}$. It remains to show that $\mathrm{cost}(P) \leqslant (1 + \frac{\varepsilon}{2})\mathsf{OPT}$.

By Lemma 2.1, there is an $(1 + \varepsilon/6)$-approximate shortest polygonal path Q_0 from s to t in $B(s, \frac{4\rho r_{a+1}}{3}) \cap |\mathcal{T}|$. Cut Q_0 at its last entry point q into $A(v_a)|_t$ to obtain a subpath Q_1 inside $A(v_a)|_t$. Without loss of generality, we make q a node of Q_1, if necessary. We apply the perturbation φ to the nodes of Q_1 and denote the result by $\varphi(Q_1)$ for convenience. Then, we augment $\varphi(Q_1)$ by the segment $\overline{s\,\varphi(q)}$. This gives us a path from s to $\varphi(t)$ in $|T(v_a)|$. Therefore, $\mathrm{cost}(P) \leqslant (1 + \frac{\varepsilon}{6})(\mathrm{cost}(\varphi(Q_1)) + \rho\,\|s\,\varphi(q)\|)$.

By Lemma 2.1, the number of nodes in Q is no more than k_ε. So the number of nodes in Q_1 is at most $k_\varepsilon + 1$ due to the possible insertion of q as a node. By the triangle inequality and Lemma 5.4, we get $\mathrm{cost}(\varphi(Q_1)) \leqslant \mathrm{cost}(Q_0) + (k_\varepsilon + 1)\pi\rho\|sv_a\|_{\mathcal{T}}$. Also, $\rho\,\|s\,\varphi(q)\| = 2\rho^2 r_a = \rho\|sv_a\|_{\mathcal{T}}$. Therefore, $\mathrm{cost}(P) \leqslant (1 + \frac{\varepsilon}{6})(\mathrm{cost}(Q_0) + (\pi(k_\varepsilon + 1) + 1)\rho\|sv_a\|_{\mathcal{T}})$. By assumption, $\mathrm{cost}(Q_0) \leqslant (1 + \frac{\varepsilon}{6})\mathsf{OPT}$. As before, we choose c in the definition of R_a so that $(\pi(k_\varepsilon + 1) + 1)\rho\|sv_a\|_{\mathcal{T}} \leqslant \frac{\varepsilon}{12} \cdot \frac{3R_a}{4\rho} < \frac{\varepsilon}{12}\|st\|_{\mathcal{T}} \leqslant \frac{\varepsilon}{12}\mathsf{OPT}$. In all, $\mathrm{cost}(P) \leqslant (1+\frac{\varepsilon}{6})(1+\frac{\varepsilon}{6}+\frac{\varepsilon}{12})\mathsf{OPT} \leqslant (1+\frac{\varepsilon}{2})\mathsf{OPT}$ for small ε.

Lemma 5.5 *Let v_i be a vertex of $\mathcal{T}$ such that $\frac{3R_i}{4\rho} < r_{i+1}$. There is a data structure $\mathsf{Gap}(v_i, \varepsilon)$ to answer $(1 + \varepsilon)$-approximate shortest path queries for any $t \in |\mathcal{T}|$ such that $\frac{3R_i}{4\rho} < \|st\|_{\mathcal{T}} < r_{i+1}$. The performance is the same as in Lemma 4.3.*

5.3 Summary

We summarize the algorithm and the results. Let $s = v_0, v_1, v_2, v_3, \ldots$ be the vertices of $\mathcal{T}$ sorted in non-decreasing geodesic distances from s. We construct the data structures $\mathsf{CloseRange}(\varepsilon)$, $\mathsf{PathQuery}(r_i, R_i, \varepsilon)$ for each v_i, and $\mathsf{Gap}(v_i, \varepsilon)$ for each v_i such that $\frac{3R_i}{4\rho} < r_{i+1}$. Given a query destination t, we obtain its geodesic distance from s in $O(\log n)$ time using the data structure in [6], which uses $O(n \log n)$ space and can be built in $O(n \log n)$ time. If $\|st\|_{\mathcal{T}} < \frac{\ell_s}{2\rho} = r_1$, we query $\mathsf{CloseRange}(\varepsilon)$. Otherwise, we locate the index a such that $r_a \leqslant \|st\|_{\mathcal{T}} < r_{a+1}$. If $r_a \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{3R_a}{4\rho}$, we query $\mathsf{PathQuery}(r_a, R_a, \varepsilon)$. Otherwise, $\frac{3R_a}{4\rho} < \|st\|_{\mathcal{T}} < r_{a+1}$ and we query $\mathsf{Gap}(v_a, \varepsilon)$. The performance follows from Lemmas 3.2, 4.3, and 5.5.

Theorem 1 *There is a data structure such that for any $t \in |\mathcal{T}|$, an $(1+\varepsilon)$ approximation of the shortest path cost from a fixed source to t can be reported in $O(\log \frac{\rho n}{\varepsilon})$ time. The approximate path can be output in time linear in its complexity, which is $O(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon})$. The data structure uses $O(\frac{\rho^2 n^4}{\varepsilon^2} \log \frac{\rho n}{\varepsilon})$ space and can be built in $O(\frac{\rho^2 n^4}{\varepsilon^2} (\log \frac{\rho n}{\varepsilon})^2)$ time.*

References

[1] L. Aleksandrov, H. Djidjev, H. Guo, A. Maheshwari, D. Nussbaum, and J.-R. Sack. Approximate shortest path queries on weighted polyhedral surfaces. In *Proc. 31st MFCS*, 2006.

[2] L. Aleksandrov, M. Lanthier, A. Maheshwari, and J.-R. Sack. An ε-approximation algorithm for weighted shortest path queries on polyhedral surfaces. In *Proc. 14th Euro CG Barcelona*, pages 19–21, 1998.

[3] 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.

[4] S.-W. Cheng, H.-S. Na, A. Vigneron, and Y. Wang. Approximating shortest paths in anisotropic regions. In *Proc. 18th annual ACM-SIAM symposium on Discrete Algorithm*, 2007.

[5] P. Chew and R. Dyrsdale. Voronoi diagrams based on convex distance functions. In *Proc. 1st Annual Symposium on Computational geometry*, pages 235–244, 1985.

[6] J. Hershberger and S. Suri. An optimal algorithm for euclidean shortest paths in the plane. *SIAM Journal on Computing*, 28:2215–2256, 1999.

[7] R. Klein and D. Wood. Voronoi Diagrams Based on General Metrics in the Plane. pages 281–291. Springer-Verlag London, UK, 1988.

[8] M. Lanthier, A. Maheshwari, and J. R Sack. Approximating Shortest Paths on Weighted Polyhedral Surfaces. *Algorithmica*, 30(4):527–562, 2001.

[9] L. Ma. *Bisectors and Voronoi Diagrams for Convex Distance Functions*. PhD thesis, FernUniversitat Hagen, 2000.

[10] 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.

[11] 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.

[12] J. Reif and Z. Sun. Movement planning in the presence of flows. *Algorithmica*, 39(2), 2004.

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