Querying Approximate Shortest Paths in Anisotropic Regions*

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

July 29, 2009

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. Let ε be any number strictly between 0 and 1. Our data structure returns a $(1+\varepsilon)$ approximation of the shortest path cost from the fixed source to a query destination in $O(\log \frac{\rho n}{\varepsilon})$ time. Afterwards, a $(1+\varepsilon)$-approximate shortest path can be reported in time $O(\log n)$ plus the complexity of the path. The data structure uses $O(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon})$ space and can be built in $O(\frac{\rho^2 n^3}{\varepsilon^2}(\log \frac{\rho n}{\varepsilon})^2)$ time. Our time and space bounds do not depend on any other parameter; in particular, they do not depend on any geometric parameter of the subdivision such as the minimum angle.

*Research of Cheng and Wang partly supported by Research Grant Council, Hong Kong , China (project no. 612107). Research of Vigneron partly supported by a Marie Curie international reintegration grant. Work done while Yajun Wang was at the Department of Computer Science and Engineering, HKUST, Hong Kong. A preliminary version of this paper appeared in the Proceedings of the 23rd Annual Symposium on Computational Geometry, 2007, 84–91.

†Department of Computer Science and Engineering, HKUST, Hong Kong. Email: scheng@cse.ust.hk

‡School of Computing, Soongsil University, Seoul, Korea. Supported by Soongsil University Research Fund. Email: hsnaa@ssu.ac.kr

§INRA, UR 341 Mathématiques et Informatique Appliquées, 78350 Jouy-en-Josas, France. Email: antoine.vigneron@jouy.inra.fr

¶Microsoft Research Asia, Beijing, China. Email: yajunw@microsoft.com

1 Introduction

The problem of computing a shortest path between two points in planar subdivisions or polyhedral surfaces arises naturally in geographic information systems, VLSI design, logistics, and motion planning. Extensive research has been conducted for the case of Euclidean path costs and for some cases of non-Euclidean path costs. In this paper, we consider the problem of constructing a data structure for a planar subdivision in order to answer approximate shortest path queries from a fixed source. We adopt a very general cost model in which distances in the faces of the subdivision are measured using possibly asymmetric convex distance functions. This model covers the Euclidean case and other non-Euclidean cases studied before.

We briefly survey some algorithmic results in the literature for the Euclidean shortest path problem. The readers are referred to a more detailed survey by Mitchell [16]. Let n denote the number of vertices of the subdivision. Hershberger and Suri [13] presented an algorithm to compute the Euclidean shortest path in the plane in $O(n \log n)$ time and $O(n \log n)$ space. Recently, Schreiber and Sharir [19] extended this result to the surface of a convex polyhedron. Har-Peled [11] presented an algorithm to compute a $(1+\varepsilon)$-approximate shortest path between two points on a convex polyhedron in $O(n + (\log n)/\varepsilon^{1.5} + 1/\varepsilon^3)$ time. The *weighted region problem* [17] is a well studied shortest path problem in the case of non-Euclidean path costs. In this setting, each face f of the subdivision is associated with a weight w_f. The cost of a subpath within f is the length of this subpath multiplied by w_f. Mitchell and Papadimitriou [17] presented the first approximation algorithm for the weighted region problem in the plane. It runs in $O(n^8 L)$ time, where L represents the maximum number of bits used in representing any input number, including the weights and the integer coordinates of the vertices of the subdivision. Later, Aleksandrov et al. [3] and Sun and Reif [20] gave algorithms that run in time linear in n. However, their running times depend on the minimum angle of the subdivision (and the weights too in the algorithm by Aleksandrov et al. [3]). Reif and Sun [18] generalized the weighted region problem to allow a uniform flow in each face and gave an algorithm whose running time depends on n and some geometric parameters of the environment.

In our previous work [6], we studied the approximate shortest path problem in the plane with a very general non-Euclidean path cost. Within each face of the subdivision, distances are measured using a possibly asymmetric convex distance function, whose unit disk is contained in a concentric Euclidean unit disk, and contains a concentric Euclidean disk with radius $1/\rho$ for some $\rho \geqslant 1$. This model includes the Euclidean case, the weighted region problem, and its extension with uniform flows. We developed an algorithm to compute, for any $\varepsilon \in (0,1)$, a $(1+\varepsilon)$-approximate shortest path in $O\left(\frac{\rho^2 n^3}{\varepsilon^2}(\log \rho) \log \frac{\rho n}{\varepsilon}\right)$ time.[1] Most notably, the running time does not depend on any geometric parameter such as the minimum angle in the subdivision.

There are two settings for the query version of the problem. First, the source is fixed and the query specifies the destination. This is usually known as the fixed source query problem. Second, the query specifies both the source and the destination. This is usually known as the two-point query problem.

In the case of Euclidean path costs, the single-source algorithm in the plane by Hershberger and Suri [13] produces a shortest path map for the fixed source. So point location can be employed to return the shortest path cost for a query destination in $O(\log n)$ time. Chiang and Mitchell [8] developed a method to answer two-point queries in the plane in $O(\log n)$ time, using roughly $O(n^{10})$ space. They also obtained various tradeoffs between preprocessing time and space. Given a real number $\varepsilon \in (0,1]$ and a fixed source on a polyhedral surface, Har-Peled [12]

[1] Although the shortest path exists, it may not be polygonal and its complexity is unknown [6].

constructed a data structure to return a $(1+\varepsilon)$-approximate shortest path cost for a query destination in $O(\log \frac{n}{\varepsilon})$ time, where the preprocessing time is $O(n^2 \log n + \frac{n}{\varepsilon} \log \frac{1}{\varepsilon} \log \frac{n}{\varepsilon})$. In the case of the weighted region problem on polyhedral surfaces, Lanthier et al. [14] presented a data structure that can answer a two-point query in $O(\log n)$ time. The approximation error is additive and it is $\Theta(WL)$, where W is the maximum weight of the faces and L is the length of the longest edge in the surface. Aleksandrov et al. [2] presented improved results for answering fixed-source and two-point queries on weighted polyhedral surfaces of arbitrary genus. Both the preprocessing time and the space depend on the weights and the minimum angle in the polyhedral surface.

In this paper, we present a data structure to answer approximate shortest path queries from a fixed source in a planar subdivision. Distances in a face are measured using a possibly asymmetric convex distance function and the convex distance functions can be different for different faces. Given a real number $\varepsilon \in (0,1)$, we can construct a data structure that reports a $(1+\varepsilon)$-approximate shortest path cost for a query destination in $O\left(\log \frac{\rho n}{\varepsilon}\right)$ time. Afterwards, a $(1+\varepsilon)$-approximate shortest path can be output in time $O(\log n)$ plus the complexity of the path, which is $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon}\right)$. The data structure can be constructed in $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \left(\log \frac{\rho n}{\varepsilon}\right)^2\right)$ time using $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon}\right)$ space. Our time and space bounds do not depend on any geometric parameter of the subdivision such as the minimum angle.

We use a variant of the discretization of the planar subdivision proposed in our previous work [6]. We face two difficulties in the query setting. First, when the query destination is very close to the fixed source, it is unclear how to efficiently obtain a good lower bound on the shortest path cost. 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. Our contribution is to introduce new techniques to overcome these difficulties. We introduce a scaling transformation and a subdivision perturbation technique. They map the query destinations in the above difficult cases to certain canonical positions. Then we build a data structure to answer the queries at canonical positions.

The preprocessing time and the space of our data structure are smaller than the results in the conference version [5] by a factor n. This is achieved by a refined analysis of the complexity of a $(1+\varepsilon)$-approximate shortest path. It brings the construction time of our query data structure to within a factor $(\log \frac{\rho n}{\varepsilon})/\log \rho$ from the running time of our previous algorithm [6] for a single approximate shortest path computation. In other words, an extra logarithmic factor in the preprocessing time allows us to answer approximate shortest path queries in $O(\log \frac{\rho n}{\varepsilon})$ time.

2 Preliminaries

Environment. We denote the input planar subdivision by $\mathcal{T}$. Some regions in $\mathcal{T}$ represent obstacles and are inaccessible. The other regions are known as the *faces* of $\mathcal{T}$. The *underlying space* $|\mathcal{T}| \subset \mathbb{R}^2$ is the union of all faces. Without loss of generality, we assume that the faces are triangles and $|\mathcal{T}|$ is connected. 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

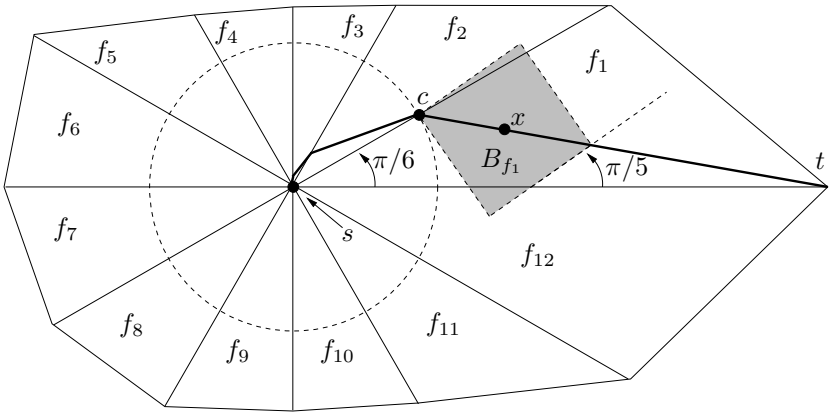

Figure 1: All face angles at s are $\pi/6$. B_{f_1} is a tilted square centered at the origin with edge length $\sqrt{2}$. $B_{f_{i+1}}$ is obtained by rotating B_{f_i} in the counterclockwise direction by $\pi/6$.

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, denoted $\|pq\|_{\mathcal{T}}$, is the Euclidean length of the shortest polyline in $|\mathcal{T}|$ with endpoints p and q. We use $B(x, r)$ to denote the closed Euclidean disk centered at x with radius r. We use $\text{int}(\cdot)$ and $\text{bd}(\cdot)$ 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 $\text{int}(\overline{pq}) \subset \text{int}(f)$ for some face f, the cost of $\overline{pq}$ is defined as $\text{cost}(\overline{pq}) = d_f(p, q)$. If $\overline{pq}$ is contained in an edge e of $\mathcal{T}$ that is adjacent to exactly one face f, we also define $\text{cost}(\overline{pq})$ to be $d_f(p, q)$. If e is adjacent to two faces f_1 and f_2, we define $\text{cost}(\overline{pq})$ 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. It means that the speed allowed in any direction belongs to the interval $[1/\rho, 1]$. Hence, $\|pq\| \leqslant \text{cost}(\overline{pq}) \leqslant \rho \|pq\|$.

Polygonal paths. We use s to denote the fixed source and t to denote the query destination. Without loss of generality, we assume that s is a vertex of $\mathcal{T}$, which can be enforced by splitting the triangle 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 the 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 $\text{length}(P) = \sum_{i=1}^{m} \|p_{i-1}p_i\|$. For any 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. Our definition of k-link paths is slightly more general than that given in [6] because we do not restrict t to be a vertex of $\mathcal{T}$. Nevertheless, the results in [6] still hold by conceptually inserting t as a vertex.

Approximation. 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. In our previous work [6], we showed that the shortest path may not be polygonal and that the shortest path may intersect some edge incident to s an infinite number of times. Figure 1 is an example such that the shortest path from s to t is not a polygonal path. In the figure, traveling directly from s to x is more expensive than traveling from s to c and then x. Repeating this argument shows that the shortest path may intersect some edge incident to s an infinite number of times. Even though a shortest path may not be polygonal, we proved in [6] that such a path exists, and thus the following quantity is well defined:

Definition 1 *For any point $t \in |\mathcal{T}|$, OPT_{st} denotes the shortest path cost from s to t in $|\mathcal{T}|$.*

A $(1+\varepsilon)$-*approximate shortest path* is a path from s to t with cost at most $(1+\varepsilon)\mathsf{OPT}_{st}$. We also proved the following results.

Lemma 2.1 ([6]) *For any $\varepsilon \in (0,1)$ and for any point $t \in |\mathcal{T}|$, the following hold:*

(i) *There is a $(1+\varepsilon)$-approximate shortest polygonal path from s to t that is a $(21\rho n^2/\varepsilon)$-link path.*

(ii) *If P is a $(1+\varepsilon)$-approximate shortest polygonal path from s to t, then $\mathrm{cost}(P) \leqslant 2\rho \, \|st\|_{\mathcal{T}}$ and $P \subset B(s, 2\rho \, \|st\|_{\mathcal{T}})$.*

For any query destination $t \in |\mathcal{T}|$, we define

$$V_{st}^{\varepsilon} = \left\{ \text{vertex } v \text{ of } \mathcal{T} : \frac{\varepsilon}{8\rho n} \|st\|_{\mathcal{T}} < \|sv\|_{\mathcal{T}} < 5\rho \|st\|_{\mathcal{T}} \right\}.$$

We can refine the complexity bound of the approximate path as follows:

Lemma 2.2 *For any $\varepsilon \in (0,1)$ and for any point $t \in |\mathcal{T}|$, there is a k_{st}^{ε}-link $(1+\varepsilon)$-approximate shortest path from s to t, where $k_{st}^{\varepsilon} = 3n(|V_{st}^{\varepsilon}| + 2)(40\rho/\varepsilon + 5) + 9n$.*

The proof of Lemma 2.2 is given in Appendix A. Notice that Lemma 2.2 is not an improvement of Lemma 2.1(i) because we can only bound $|V_{st}^{\varepsilon}|$ by n and get the same bound of $O(\rho n^2/\varepsilon)$. However, when we have a number of destinations $t_1, t_2, \ldots, t_m$, we can show a tighter bound of $\sum_i |V_{st_i}^{\varepsilon}|$ than $O(mn)$ under certain conditions. This allows us to improve the results in the conference version [5] by a factor of n.

Preprocessing. We need to split some edges of $\mathcal{T}$ to enforce a structural property. For any edge e of $\mathcal{T}$, if s projects orthogonally onto the interior of e, we insert the projection of s as a new vertex and split e into two shorter edges. As a consequence, for any edge of the subdivision, the nearest point from s to the edge is an endpoint. For convenience, we still use $\mathcal{T}$ to denote the modified subdivision and use n to denote the number of vertices in the modified subdivision which is within a constant factor of the number of input vertices. The count n includes the source s, so there are $n-1$ vertices other than s. Clearly, Lemma 2.1 and Lemma 2.2 still hold with this larger n.

Additively weighted Voronoi diagrams. We will put weighted Steiner points on the edges of $\mathcal{T}$ so that ideally, for a query point t in a face f, we can find the Steiner point $s_i \in \mathrm{bd}(f)$ that minimizes $d_f(s_i, t) + w_i$, where w_i is the approximate shortest path cost from s to s_i.

Thus the Steiner points in $\mathrm{bd}(f)$ act as gateways through which an approximate shortest path from s to t may enter f. This motivates us to employ the *additively weighted Voronoi diagram* under the convex distance function d_f. The difficulty is that while we assume that $d_f(x, y)$ can be evaluated in $O(1)$ time given two points x and y, we do not know how to compute the bisector of two points under d_f without further information about B_f. Therefore, for each face f, we approximate d_f by computing a convex polygon approximating B_f. Then, we can use the additively weighted Voronoi diagram under the approximation of d_f. Specifically, we have the following two results whose proofs are given in Appendix B.

Lemma 2.3 *For any $\varepsilon \in (0, 1)$ and for any face f of $\mathcal{T}$, we can compute a convex polygon $C_f^\varepsilon \subseteq B_f$ in $O(\rho/\varepsilon)$ time, such that:*

(i) *C_f^ε has $O(\rho/\varepsilon)$ vertices.*

(ii) *Let d_f^ε be the convex distance function induced by C_f^ε. For any two points x and y in $\mathbb{R}^2$, $d_f(x, y) \leqslant d_f^\varepsilon(x, y) \leqslant (1 + \varepsilon)d_f(x, y)$.*

Lemma 2.4 *Let d_f^ε be the convex distance function defined in Lemma 2.3 for a face f of $\mathcal{T}$ and for some $\varepsilon \in (0, 1)$. Let $S = \{s_1, s_2, \ldots\}$ be a set of point sites such that each site s_i is associated with a non-negative weight w_i and $w_i \leqslant w_j$ for $i < j$. Then the additively weighted Voronoi diagram $\mathrm{Vor}(S, d_f^\varepsilon)$ of S under d_f^ε satisfies the following:*

(i) *If the Voronoi cell V_i of s_i is non-empty, then V_i is a star-shaped polygon and $s_i \in \mathrm{int}(V_i)$.*

(ii) *When the sites of S lie on a line, the complexity of $\mathrm{Vor}(S, d_f^\varepsilon)$ is $O\left(\frac{\rho}{\varepsilon}|S|\right)$ and $\mathrm{Vor}(S, d_f^\varepsilon)$ can be constructed in $O\left(\frac{\rho}{\varepsilon}|S| \log |S|\right)$ time.*

(iii) *For any point $x \in f$, we have $\min_{s_i \in S}\{w_i + d_f^\varepsilon(s_i, x)\} \leqslant (1 + \varepsilon) \min_{s_i \in S}\{w_i + d_f(s_i, x)\}$.*

By Lemma 2.3 and Lemma 2.4, given a set S of weighted Steiner points on the boundary of a face f, we can compute d_f^ε and three additively weighted Voronoi diagrams, one for the weighted Steiner points on each edge of f, in $O\left(\frac{\rho}{\varepsilon}|S| \log |S|\right)$ time. Afterwards, Lemma 2.4(iii) implies that locating the query destination t in the three Voronoi diagrams allows us to approximate $\min_{s_i \in S}\{w_i + d_f(s_i, x)\}$ within a factor of $1 + \varepsilon$. This is sufficient for our purposes.

3 A building block

Let r and R be two real numbers such that $r \leqslant R/(2\rho)$. We describe a data structure that answers approximate shortest path queries for destinations t such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant R/(2\rho)$. This data structure is a building block for our final query data structure. We need some definitions first.

Definition 2 *For any $R \geqslant r > 0$ and for any $\varepsilon \in (0, 1)$, define*

$$V_{r,R}^\varepsilon = \left\{ \text{vertex } v \text{ of } \mathcal{T} : \frac{\varepsilon r}{8\rho n} < \|sv\|_{\mathcal{T}} < \frac{5}{2}R \right\}$$

$$k_{r,R}^\varepsilon = 3n(|V_{r,R}^\varepsilon| + 2)\,(40\rho/\varepsilon + 5) + 9n.$$

Notice the similarity between V_{st}^ε and $V_{r,R}^\varepsilon$ and the similarity between k_{st}^ε and $k_{r,R}^\varepsilon$. Indeed, if $r \leqslant \|st\|_{\mathcal{T}} \leqslant R/(2\rho)$, then $V_{st}^\varepsilon \subseteq V_{r,R}^\varepsilon$ and $k_{st}^\varepsilon \leqslant k_{r,R}^\varepsilon$. The next lemma follows from Lemma 2.2.

Lemma 3.1 *Let r and R be two parameters such that $0 < r \leqslant R/(2\rho)$. Let t be a point in $|\mathcal{T}|$ such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant R/(2\rho)$. For any $\varepsilon \in (0,1)$, there exists a $(1+\varepsilon)$-approximate shortest path from s to t that is a $k^{\varepsilon}_{r,R}$-link path.*

3.1 Data structure

First, we discretize the edges of $\mathcal{T}$. For each edge e of $\mathcal{T}$ and for $0 \leqslant i \leqslant \lceil \log(R/r) \rceil$, we insert the intersection points between e and the boundary of $B(s, 2^i r)$, and then place a maximal set of Steiner points on $\mathrm{int}(e \cap (B(s,2^i r) \setminus B(s, 2^{i-1} r)))$ with uniform spacing $2^i \varepsilon r/(24\rho k^{\omega}_{r,R})$, where $\omega = \varepsilon/8$. Let N be the point set consisting of the Steiner points created above and the vertices of $\mathcal{T}$ in $B(s, 2^{\lceil \log(R/r) \rceil} r)$. There are no more than $3n$ edges in $\mathcal{T}$. Thus,

$$|N| \leqslant \frac{144\rho n}{\varepsilon} \cdot k^{\omega}_{r,R} \cdot \left\lceil \log \frac{R}{r} \right\rceil = O\left(\frac{\rho^2 n^2}{\varepsilon^2} \left(|V^{\omega}_{r,R}| + 2 \right) \log \frac{R}{r} \right).$$

Second, let G be a weighted directed graph with vertex set N such that any two points $p_i, p_j \in N$ lying on the boundary of the same face are connected by an edge with weight $\mathrm{cost}(\overline{p_i p_j})$. Using the BUSHWHACK algorithm [20], we can compute the shortest paths in G from s to all other vertices of G without constructing the edges of G. For each $p_i \in N$, we assign the cost of the shortest path from s to p_i in G as the weight $w(p_i)$ of p_i.

Finally, for each face f of $\mathcal{T}$ and for each edge of f, we construct an additively weighted Voronoi diagram of the Steiner points on this edge, with the weights assigned above and under the distance function d^{ε}_f. By Lemma 2.3 and Lemma 2.4, this can be done over all faces in time $O\left(\frac{\rho}{\varepsilon}|N| \log |N|\right) = O\left(\frac{\rho^3 n^2}{\varepsilon^3}\left(|V^{\omega}_{r,R}| + 2\right)\left(\log \frac{R}{r}\right) \log |N|\right)$. We can improve the complexity by a factor of ρ/ε, using the following observation that we do not need the Voronoi diagram for all Steiner points on an edge. Take a face f and its edge e. For each integer $0 \leqslant i \leqslant \lceil \log(R/r) \rceil$, we regard every $k^{\omega}_{r,R}$ consecutive Steiner points in $e \cap (B(s,2^i r) \setminus B(s, 2^{i-1} r))$ as a cluster. In each cluster, the Steiner point p_j with minimum weight $w(p_j)$ is assigned as the *cluster hub*. In the next subsection, we will show that the additively weighted Voronoi diagram of the cluster hubs in e already serves our purposes. This improvement saves a factor of ρ/ε in the time and space complexities of our data structure, since the two complexities are dominated by the running time of BUSHWHACK and the number of Steiner points placed in $\mathcal{T}$. (See the proof of Lemma 3.3 for more detailed analysis.)

Using the data structure described above, we can answer the query for a destination t such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant R/(2\rho)$ as follows. 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 of cluster hubs associated with the edges of f. (The face f has three sides and each side may be split into two edges in the preprocessing. So there are at most six Voronoi diagrams.) This gives us the cluster hub, denoted h, that achieves $\min\{\, w(p_i) + d^{\omega}_f(p_i, t) \,:\, p_i \in \mathrm{bd}(f) \text{ is a cluster hub} \,\}$. We report $w(h) + d_f(h,t)$ as the approximate path cost and the precomputed shortest path in G from s to h followed by $\overline{ht}$ as the approximate shortest path from s to t. Analysis of the time and space complexities of all constructions and querying is left to Lemma 3.3.

3.2 Analysis

The following lemma shows that our data structure answers queries correctly.

Lemma 3.2 *Let r and R be two real numbers such that $0 < r \leqslant R/(2\rho)$. Let t be a point in $|\mathcal{T}|$ such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant R/(2\rho)$. Let f be the face of $\mathcal{T}$ containing t. Let $h \in \mathrm{bd}(f)$ be*

the cluster hub that minimizes $w(p) + d_f^\omega(p,t)$ over all cluster hubs p in $\mathrm{bd}(f)$, where $\omega = \varepsilon/8$. Then $w(h) + d_f(h,t) \leqslant (1+\varepsilon)\mathsf{OPT}_{st}$.

Proof. Let Q be a $k_{r,R}^\omega$-link $(1+\omega)$-approximate shortest path from s to t. The existence of Q is guaranteed by Lemma 3.1. By Lemma 2.1(ii) and the fact that $\mathrm{length}(Q) \leqslant \mathrm{cost}(Q)$, we have $\mathrm{length}(Q) \leqslant \mathrm{cost}(Q) \leqslant 2\rho\,\|st\|_{\mathcal{T}}$. Since $\|st\|_{\mathcal{T}} \leqslant R/(2\rho)$, we have $\mathrm{length}(Q) \leqslant R$ and thus $Q \subset B(s,R) \subseteq B(s, 2^{\lceil \log(R/r)\rceil} r)$. So Q lies inside the portion of $\mathcal{T}$ that has been discretized.

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 except s, x, and t as follows. Assume that $w \in e \cap (B(s,2^i r) \setminus B(s,2^{i-1}r))$ for some edge e of $\mathcal{T}$ and for some $0 \leqslant i \leqslant \lceil \log(R/r)\rceil$. By construction, there is a Steiner point on e at distance at most $2^i\varepsilon r/(24\rho k_{r,R}^\omega)$ from w. We snap w to this Steiner point and the path cost increases by at most $2^i\varepsilon r/(12k_{r,R}^\omega)$. Observe that $\mathrm{cost}(Q) \geqslant \mathrm{length}(Q) \geqslant \|sw\|_{\mathcal{T}} \geqslant 2^{i-1}r$ and thus the additional cost caused by this snapping is at most $\varepsilon\,\mathrm{cost}(Q)/(6k_{r,R}^\omega)$. Since Q has fewer than $k_{r,R}^\omega$ nodes other than s, t and x, the total extra cost is at most $\frac{\varepsilon}{6}\mathrm{cost}(Q)$.

We snap x to the nearest cluster hub p_j on the edge of f containing x. Let i be the integer such that $x \in B(s,2^i r) \setminus B(s,2^{i-1}r)$. So $\|xp_j\| \leqslant 2^i\varepsilon r/(24\rho)$. Since $\mathrm{cost}(Q) \geqslant \mathrm{length}(Q) \geqslant \|sx\|_{\mathcal{T}} \geqslant 2^{i-1}r$, we get $\|xp_j\| \leqslant \frac{\varepsilon}{12\rho}\mathrm{cost}(Q)$. So the cost of this detour is at most $2\rho\,\|xp_j\| \leqslant \frac{\varepsilon}{6}\mathrm{cost}(Q)$.

In total, this modified path from s to t through the cluster hub p_j has cost at most $\mathrm{cost}(Q) + \frac{\varepsilon}{3}\mathrm{cost}(Q) \leqslant (1+\frac{\varepsilon}{3})(1+\omega)\mathsf{OPT}_{st} \leqslant (1+\frac{\varepsilon}{2})\mathsf{OPT}_{st}$. Also, since any path from s to p_j in G has cost at least $w(p_j)$, the cost of this modified path must be at least $w(p_j) + d_f(p_j,t)$. Thus we get $w(p_j) + d_f(p_j,t) \leqslant (1+\frac{\varepsilon}{2})\mathsf{OPT}_{st}$. By Lemma 2.3(ii), we have $w(h) + d_f(h,t) \leqslant w(h) + d_f^\omega(h,t) \leqslant w(p_j) + d_f^\omega(p_j,t) \leqslant (1+\omega)(w(p_j) + d_f(p_j,t)) \leqslant (1+\omega)(1+\frac{\varepsilon}{2})\mathsf{OPT}_{st} \leqslant (1+\varepsilon)\mathsf{OPT}_{st}$. □

The following lemma summarizes the performance of the data structure.

Lemma 3.3 *For any r and R such that $0 < r \leqslant R/(2\rho)$ and for any $\varepsilon \in (0,1)$, we can set $\omega = \varepsilon/8$ and construct a data structure $\mathsf{PathQuery}(\mathcal{T}, r, R, \varepsilon)$ such that:*

(i) *For any point $t \in |\mathcal{T}|$ such that $r \leqslant \|st\|_{\mathcal{T}} \leqslant R/(2\rho)$,*

- *the cost of a $(1+\varepsilon)$-approximate shortest path from s to t can be reported in $O(\log\frac{\rho n}{\varepsilon} + \log\log\frac{R}{r})$ time;*
- *afterwards, a $(1+\varepsilon)$-approximate shortest polygonal path can be output in time linear in its complexity, and the complexity of this path is at most*

$$\frac{144\rho n}{\varepsilon} \cdot k_{r,R}^\omega \cdot \left\lceil \log\frac{R}{r} \right\rceil = O\left(\frac{\rho^2 n^2}{\varepsilon^2}\left(|V_{r,R}^\omega| + 2\right)\log\frac{R}{r}\right).$$

(ii) *It uses $O\left(\frac{\rho^2 n^2}{\varepsilon^2}(|V_{r,R}^\omega| + 2)\log\frac{R}{r}\right)$ space.*

(iii) *It can be constructed in $O\left(\frac{\rho^2 n^2}{\varepsilon^2}(|V_{r,R}^\omega| + 2)\left(\log\frac{R}{r}\right)\left(\log\frac{\rho n}{\varepsilon} + \log\log\frac{R}{r}\right)\right)$ time.*

Proof. The correctness of the answer for the query has been proved in Lemma 3.2. The query algorithm involves a point location in $\mathcal{T}$ for the face f containing t and the point locations in the Voronoi diagrams of the cluster hubs on $\mathrm{bd}(f)$. The first point location takes $O(\log n)$ time [9]. The number of cluster hubs on an edge of $\mathrm{bd}(f)$ is $O\left(\frac{\rho}{\varepsilon}\log\frac{R}{r}\right)$, so by Lemma 2.4, each

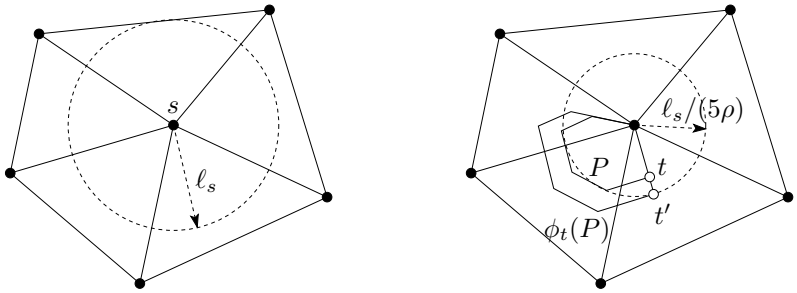

Figure 2: Scaling transformation.

Voronoi diagram has complexity $O(\frac{\rho^2}{\varepsilon^2}\log\frac{R}{r})$. It follows that the point locations in the Voronoi diagrams take $O(\log\frac{\rho}{\varepsilon}+\log\log\frac{R}{r})$ time [9]. The returned $(1+\varepsilon)$-approximate shortest path has complexity at most $|N| = O\left(\frac{\rho^2 n^2}{\varepsilon^2}(|V^{\omega}_{r,R}|+2)\log\frac{R}{r}\right)$.

We now analyze the space and time complexities of constructing $\mathsf{PathQuery}(\mathcal{T}, r, R, \varepsilon)$. Computing the shortest path tree in G rooted at s by the BUSHWHACK algorithm [20] requires $O(|N|\log|N|) = O\left(\frac{\rho^2 n^2}{\varepsilon^2}(|V^{\omega}_{r,R}|+2)(\log\frac{R}{r})(\log\frac{\rho n}{\varepsilon}+\log\log\frac{R}{r})\right)$ time and $O(|N|) = O\left(\frac{\rho^2 n^2}{\varepsilon^2}(|V^{\omega}_{r,R}|+2)\log\frac{R}{r}\right)$ space.

The point location structure for $\mathcal{T}$ takes $O(n)$ space and can be built in $O(n\log n)$ time [9]. By Lemma 2.4, the construction of the Voronoi diagrams for a face requires $O(\frac{\rho}{\varepsilon}H)$ space and $O(\frac{\rho}{\varepsilon}H\log H)$ time, where H is the number of cluster hubs in a face. Since $H = O\left(\frac{\rho}{\varepsilon}\log\frac{R}{r}\right)$ and there are $O(n)$ faces, we need $O\left(\frac{\rho^2 n}{\varepsilon^2}\log\frac{R}{r}\right)$ space and $O\left(\frac{\rho^2 n}{\varepsilon^2}(\log\frac{R}{r})(\log\frac{\rho}{\varepsilon}+\log\log\frac{R}{r})\right)$ time. The construction of the point location structures for these Voronoi diagrams requires the same time and space, so we are done. □

4 Close range queries

The data structure in Lemma 3.3 cannot handle a destination t that is arbitrarily close to s because our time and space bounds go to infinity when r goes to 0. Let ℓ_s be the distance from s to the nearest vertex in $\mathcal{T}$. We propose a method to handle query destinations that fall inside $B(s, \ell_s/(5\rho))$. The idea is to map such a destination t to a point t' further away from s, at which the data structure in Lemma 3.3 can be applied. The map should guarantee that the query results for t' can be transformed back to query results for t. Because we assume that $t \in B(s, \ell_s/(5\rho))$, any $(1+\varepsilon)$-approximate shortest polygonal path from s to t lies inside $B(s, \ell_s)$ (Lemma 2.1(ii)). By our preprocessing of $\mathcal{T}$ in Section 2, the closest point to s from any edge of $\mathcal{T}$ is an edge endpoint, so $B(s, \ell_s)$ lies inside the union of faces incident to s. Hence, any $(1+\varepsilon)$-approximate shortest path lies within the union of the faces incident to s. The geometry of this environment is very simple and allows us to define a scaling transformation $\phi_t : \mathbb{R}^2 \to \mathbb{R}^2$. Let t' be the intersection point between the boundary of $B(s, \ell_s/(5\rho))$ and the ray from s through t.

Definition 3 *For any point $p \in \mathbb{R}^2$, define $\phi_t(p) = s + \dfrac{\|st'\|}{\|st\|}(p - s)$.*

Under our assumptions, a path P is a $(1+\varepsilon)$-approximate shortest path from s to t if and only if $\phi_t(P)$ is a $(1+\varepsilon)$-approximate shortest path from s to t'. Figure 2 shows an example of the scaling transformation. Since $\|st'\| = \|st'\|_{\mathcal{T}} = \ell_s/(5\rho)$, we can answer the query for t' using the data structure $\mathsf{PathQuery}(\mathcal{T}, \ell_s/(5\rho), 2\ell_s/5, \varepsilon)$. Afterwards, we can report the approximate shortest path from s to t by applying ϕ_t^{-1} to the approximate shortest path from s to t' in time linear in its complexity. Notice that the set $V^{\omega}_{r,R}$ in Lemma 3.3 is empty for $r = \ell_s/(5\rho)$ and $R = 2\ell_s/5$, and thus $k^{\omega}_{r,R} = 6n(320\rho/\varepsilon + 5) + 9n \leqslant 6n(320\rho/\varepsilon + 7)$. Summarizing this, we get the following lemma.

Lemma 4.1 *For any $\varepsilon \in (0,1)$, we can construct a data structure $\mathsf{CloseRange}(\mathcal{T}, \varepsilon)$ such that:*

(i) *For any point $t \in B(s, \ell_s/(5\rho)) \cap |\mathcal{T}|$,*

- *the cost of a $(1+\varepsilon)$-approximate shortest path from s to t can be reported in $O\left(\log \frac{\rho n}{\varepsilon}\right)$ time;*
- *afterwards, a $(1+\varepsilon)$-approximate shortest polygonal path can be output in time linear in its complexity, and the complexity of this path is at most*

$$\frac{864\rho n^2}{\varepsilon}\left(\frac{320\rho}{\varepsilon} + 7\right)\lceil \log(2\rho)\rceil = O\left(\frac{\rho^2 n^2}{\varepsilon^2}\log\rho\right).$$

(ii) *It uses $O\left(\frac{\rho^2 n^2}{\varepsilon^2}\log\rho\right)$ space.*

(iii) *It can be constructed in $O\left(\frac{\rho^2 n^2}{\varepsilon^2}\log\rho\log\frac{\rho n}{\varepsilon}\right)$ time.*

The performance of the data structure above is independent of the value of ℓ_s because it is cancelled in the ratio R/r. So there is no impact even if s is close to an edge of $\mathcal{T}$ and thus ℓ_s is very small.

5 Gap query structure

Lemma 4.1 enables us to handle query destinations at distance at most $\ell_s/(5\rho)$ from s. To handle query destinations further away from s, a natural strategy is to divide the plane into geodesic annuli and construct the data structure in Lemma 3.3 for each geodesic annulus. (We use the term geodesic annulus because the "distance" of a point x from s is measured by its geodesic distance $\|sx\|_{\mathcal{T}}$ from s.)

The difficulty is that the construction time of this data structure for a geodesic annulus goes to infinity when the ratio between the two radii goes to infinity. So we may need an arbitrarily large number of such geodesic annuli in order to cover a triangulation $\mathcal{T}$ with large spread. In order to control the complexity, we do not require the geodesic annuli to cover the entire plane. This leaves some uncovered gaps (which are geodesic annuli too). In this section, we propose a data structure to handle query destinations that fall into such gaps.

Let $s = v_0, v_1, v_2, v_3, \ldots, v_{n-1}$ be the vertices of $\mathcal{T}$ sorted in order of non-decreasing geodesic distances from s. For each vertex v_i of $\mathcal{T}$, we define

$$\begin{aligned} r_i &= \frac{1}{6\rho^2}\cdot\|sv_i\|_{\mathcal{T}} \\ R_i &= C_R\cdot\frac{\rho^4 n^2}{\varepsilon^3}\cdot\log(2\rho)\cdot\|sv_i\|_{\mathcal{T}}, \end{aligned}$$

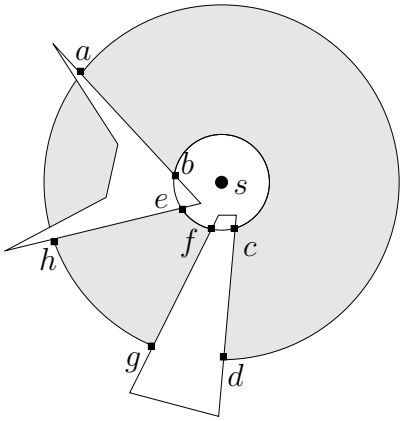

Figure 3: The white polygons are obstacles. The larger disk is $B(s, 3\rho r_a)$ and the smaller disk is $B(s, 6\rho^2 r_{a-1})$. The union of the smaller disk and the shaded regions $abcd$ and $efgh$ is $\widehat{B}(s, 3\rho r_a)$. The shaded regions $abcd$ and $efgh$ form $\widehat{A}(s, 3\rho r_a)$, which is disconnected.

where C_R is a sufficiently large constant, to be specified later. In particular, we will have $C_R > 1$, so that $0 < r_i < R_i/(2\rho)$. Therefore, if a query destination t satisfies $r_i \leqslant \|st\|_{\mathcal{T}} \leqslant R_i/(2\rho)$, this query can be handled by the data structure $\mathsf{PathQuery}(\mathcal{T}, r_i, R_i, \varepsilon)$. Since $r_1 = \|sv_1\|_{\mathcal{T}}/(6\rho^2) = \ell_s/(6\rho^2) < \ell_s/(5\rho)$, $\mathsf{CloseRange}(\mathcal{T}, \varepsilon)$ and $\mathsf{PathQuery}(\mathcal{T}, r_1, R_1, \varepsilon)$ together handle all query destinations that are within a geodesic distance of $R_1/(2\rho)$ from s. So the nearest gap from s is at a geodesic distance greater than $R_1/(2\rho)$ from s. The definition of a gap is given below.

Definition 4 *For any integer $a \in [2, n-1]$ such that $R_{a-1}/(2\rho) < r_a$, define*

$$\mathsf{Gap}(a) = \{\textit{point } x \in |\mathcal{T}| \,:\, R_{a-1}/(2\rho) < \|sx\|_{\mathcal{T}} < r_a\}\,.$$

5.1 Structural properties

We first prove some structural properties of $\mathsf{Gap}(a)$; the following definitions are needed.

Definition 5 *For any $r > 0$, $\widehat{B}(s, r)$ is the connected component of $B(s, r) \cap |\mathcal{T}|$ containing s.*

Definition 6 *For any integer $a \in [2, n-1]$, define*

$$\begin{aligned} A(s, 3\rho r_a) &= B(s, 3\rho r_a) \setminus B(s, 6\rho^2 r_{a-1}) \\ \widehat{A}(s, 3\rho r_a) &= \widehat{B}(s, 3\rho r_a) \setminus B(s, 6\rho^2 r_{a-1}). \end{aligned}$$

Note that $3\rho r_a = \|sv_a\|_{\mathcal{T}}/(2\rho)$ and $6\rho^2 r_{a-1} = \|sv_{a-1}\|_{\mathcal{T}}$. Although $\widehat{B}(s, 3\rho r_a)$ is connected, $\widehat{A}(s, 3\rho r_a)$ may be disconnected. Figure 3 shows an example. We first show that $\mathsf{Gap}(a) \subseteq \widehat{A}(s, 3\rho r_a)$. The following technical lemma will be useful in proving this statement.

Lemma 5.1 *If $R_{a-1}/(2\rho) < r_a$ for some integer $a \in [2, n-1]$, then any vertex v_i of $\mathcal{T}$ in $\widehat{B}(s, 3\rho r_a)$ satisfies $\|sv_i\|_{\mathcal{T}} \leqslant \|sv_{a-1}\|_{\mathcal{T}}$ and thus $v_i \in B(s, \|sv_{a-1}\|_{\mathcal{T}}) = B(s, 6\rho^2 r_{a-1})$.*

Proof. Since $v_i \in \widehat{B}(s, 3\rho r_a)$, there exists a polygonal path $P \subset \widehat{B}(s, 3\rho r_a)$ from s to v_i such that all nodes of P are vertices in $\mathcal{T}$. For example, take P to be the geodesic path from s to v_i in $\widehat{B}(s, 3\rho r_a)$. We prove by induction along P from s that $\|sv_k\|_{\mathcal{T}} \leqslant \|sv_{a-1}\|_{\mathcal{T}}$ for every node

v_k of P. The base case is trivial as s is the first node of P. Assume that $\|sv_j\|_{\mathcal{T}} \leqslant \|sv_{a-1}\|_{\mathcal{T}}$ for some node v_j of P. Let v_k be the next node along P. Since $P \subset \widehat{B}(s, 3\rho r_a)$, we have $\|sv_k\| \leqslant 3\rho r_a = \|sv_a\|_{\mathcal{T}}/(2\rho)$. Therefore,

$$\begin{aligned} \|sv_k\|_{\mathcal{T}} &\leqslant \|sv_j\|_{\mathcal{T}} + \|v_j v_k\| \\ &\leqslant \|sv_j\|_{\mathcal{T}} + \|sv_j\| + \|sv_k\| \\ &\leqslant 2\|sv_j\|_{\mathcal{T}} + \|sv_a\|_{\mathcal{T}}/(2\rho). \end{aligned}$$

Because $\|sv_j\|_{\mathcal{T}} \leqslant \|sv_{a-1}\|_{\mathcal{T}}$, by the inductive assumption, we obtain

$$\|sv_k\|_{\mathcal{T}} \leqslant 2\|sv_{a-1}\|_{\mathcal{T}} + \|sv_a\|_{\mathcal{T}}/(2\rho).$$

The definition of R_{a-1} implies that $2\|sv_{a-1}\|_{\mathcal{T}} \leqslant R_{a-1}$. Since we assume that $R_{a-1} < 2\rho r_a$, we get $2\|sv_{a-1}\|_{\mathcal{T}} < 2\rho r_a = \|sv_a\|_{\mathcal{T}}/(3\rho)$. Hence, $\|sv_k\|_{\mathcal{T}} < \frac{1}{\rho}\|sv_a\|_{\mathcal{T}} \leqslant \|sv_a\|_{\mathcal{T}}$. Since we assign indices to the vertices of $\mathcal{T}$ in non-decreasing order of their geodesic distances from s, we conclude that $\|sv_k\|_{\mathcal{T}} \leqslant \|sv_{a-1}\|_{\mathcal{T}}$. □

As a consequence, there is no vertex of $\mathcal{T}$ in $\widehat{A}(s, 3\rho r_a)$ for any integer $a \in [2, n-1]$.

Corollary 1 *If $R_{a-1}/(2\rho) < r_a$ for some integer $a \in [2, n-1]$, no vertex of $\mathcal{T}$ lies inside $\widehat{A}(s, 3\rho r_a)$.*

The next result shows that $\mathsf{Gap}(a) \subseteq \widehat{A}(s, 3\rho r_a)$.

Lemma 5.2 *If $R_{a-1}/(2\rho) < r_a$ for some integer $a \in [2, n-1]$, then $\mathsf{Gap}(a) \subseteq \widehat{A}(s, 3\rho r_a)$.*

Proof. Take a point $x \in \mathsf{Gap}(a)$. Since $\|sx\|_{\mathcal{T}} < r_a$, a geodesic path P from s to x in $|\mathcal{T}|$ must lie inside $\widehat{B}(s, r_a)$, which implies that $x \in \widehat{B}(s, r_a) \subset \widehat{B}(s, 3\rho r_a)$. It remains to show that $x \notin B(s, 6\rho^2 r_{a-1}) = B(s, \|sv_{a-1}\|_{\mathcal{T}})$.

Let v_i be the node preceding x in P. Since P is a geodesic path, v_i is a vertex of $\mathcal{T}$. It could be the case that $v_i = s$. Since $v_i \in \widehat{B}(s, r_a)$, by Lemma 5.1,

$$\|sv_i\|_{\mathcal{T}} \leqslant \|sv_{a-1}\|_{\mathcal{T}}.$$

We complete the proof by contradiction. So we assume that $x \in B(s, 6\rho^2 r_{a-1}) = B(s, \|sv_{a-1}\|_{\mathcal{T}})$. Then

$$\begin{aligned} \|sx\|_{\mathcal{T}} &\leqslant \|sv_i\|_{\mathcal{T}} + \|v_i x\| \\ &\leqslant \|sv_i\|_{\mathcal{T}} + \|sv_i\| + \|sx\| \\ &\leqslant 2\|sv_i\|_{\mathcal{T}} + \|sx\| \\ &\leqslant 3\|sv_{a-1}\|_{\mathcal{T}}. \end{aligned}$$

By the definition of R_{a-1}, $3\|sv_{a-1}\|_{\mathcal{T}} < R_{a-1}/(2\rho)$ and thus $\|sx\|_{\mathcal{T}} < R_{a-1}/(2\rho)$, which contradicts the assumption that $x \in \mathsf{Gap}(a)$. □

5.2 Perturbation

We need to handle query destinations that fall into $\mathsf{Gap}(a)$ for some integers $a \in [2, n-1]$. By Lemma 5.2, we can broaden our focus to $\widehat{A}(s, 3\rho r_a)$ whose geometric characterization is easier to work with. Recall that $\widehat{A}(s, 3\rho r_a) = \widehat{B}(s, 3\rho r_a) \setminus B(s, 6\rho^2 r_{a-1})$. So s is relatively far away from the inner circular boundary of $\widehat{A}(s, 3\rho r_a)$. Also, by Corollary 1, no vertex of $\mathcal{T}$ lies inside $\widehat{A}(s, 3\rho r_a)$.

Let a be any integer in the range $[2, n-1]$ such that $R_{a-1}/(2\rho) < r_a$. Take any edge e of $\mathcal{T}$ that intersects $\widehat{A}(s, 3\rho r_a)$. By Corollary 1, $\widehat{A}(s, 3\rho r_a)$ does not contain any endpoint of e. Moreover, no ball centered at s makes a tangential contact with an interior point of e. (This is the reason for inserting new vertices in the interior of input edges in the preprocessing of Section 2.) So one endpoint of e lies inside $B(s, 6\rho^2 r_{a-1})$ and the other endpoint of e lies outside $B(s, 3\rho r_a)$. See Figure 4 for an illustration of two such edges. This property allows us to order all the edges of $\mathcal{T}$ intersecting $\widehat{A}(s, 3\rho r_a)$ in clockwise order. Let $E_a = \{e_1, e_2, \ldots\}$ be a sorted list of such edges of $\mathcal{T}$.

If we ignore the portion of $|\mathcal{T}|$ inside $B(s, 6\rho^2 r_{a-1})$, the situation is similar to the case of close range queries in Section 4. Still, there is one notable difference to be handled differently: the edges of $\mathcal{T}$ that intersect $\widehat{A}(s, 3\rho r_a)$ are not incident to s. To remedy this situation, we perturb these edges so that their support lines pass through s and thus the data structure $\mathsf{CloseRange}(\mathcal{T}, \varepsilon)$ can be used. This perturbation and our neglecting the portion of $|\mathcal{T}|$ in $B(s, 6\rho^2 r_{a-1})$ incur some error. However, we will show that this error does not harm our approximation too much. We explain the perturbation and prove a technical lemma in the rest of this subsection. We describe and analyze the data structures that use this perturbation in the next subsection.

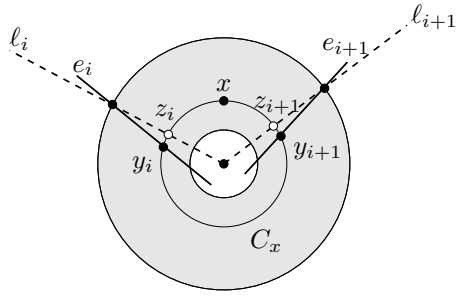

Figure 4: The inner circle denotes $B(s, 6\rho^2 r_{a-1})$. The outer circle denotes $B(s, 3\rho r_a)$. The middle circle is C_x. The perturbation defines a linear transformation from the arc $C_x[y_i, y_{i+1}]$ to the arc $C_x[z_i, z_{i+1}]$.

Now we perturb the edges in E_a so that their support lines pass through s. (See Figure 4.) Let u_i be the intersection point between e_i and the boundary of $B(s, 3\rho r_a)$. Let ℓ_i denote the ray from s through u_i, which can be viewed as a perturbation of the support line of e_i. The rays partition $\mathbb{R}^2$ into cones. We associate a convex distance function with each cone as follows. 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. As a result, the cones associated with convex distance functions form a new subdivision having only one vertex s. We denote this subdivision by $\mathcal{S}_a$. The cones in $\mathcal{S}_a$ are unbounded and incident to s.

The perturbation applies to the edges in E_a so far. We extend it to every point in $\widehat{A}(s, 3\rho r_a)$ as follows. For any point x in $\widehat{A}(s, 3\rho r_a)$, let C_x be the circle that is centered at s and passes through x. Refer to Figure 4. 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

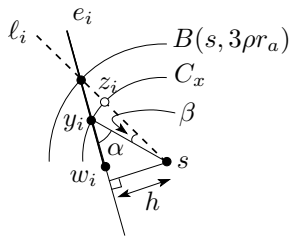

Figure 5: Because one endpoint w_i of e_i is inside $B(s, 6\rho^2 r_{a-1})$, the distance h from s to e_i is at most $6\rho^2 r_{a-1}$. The arc length between y_i and z_i is $h\beta/\sin\alpha$.

$z_{i+1} = C_x \cap \ell_{i+1}$. Given two points p and q on the arc of C_x traversed in clockwise order from y_i to y_{i+1}, denote by $C'_x[p, q]$ the subarc between p and q. Similarly, given two points p and q on the arc of C_x traversed in clockwise order from z_i to z_{i+1}, denote by $C''_x[p, q]$ the subarc between p and q. We can now define the perturbation φ_a from $\widehat{A}(s, 3\rho r_a)$ to $\mathcal{S}_a$.

Definition 7 *For any point $x \in \widehat{A}(s, 3\rho r_a)$ on $C'_x[y_i, y_{i+1}]$, define $\varphi_a(x)$ to be the point on $C''_x[z_i, z_{i+1}]$ such that*

$$\frac{\text{arclength}(C''_x[z_i, \varphi_a(x)])}{\text{arclength}(C''_x[z_i, z_{i+1}])} = \frac{\text{arclength}(C'_x[y_i, x])}{\text{arclength}(C'_x[y_i, y_{i+1}])}.$$

Note that $\varphi_a(y_i) = z_i$, $\varphi_a(y_{i+1}) = z_{i+1}$, and φ_a is a bijection between $\widehat{A}(s, 3\rho r_a)$ and $\varphi_a(\widehat{A}(s, 3\rho r_a))$. The next lemma provides an upper bound on the Euclidean distance between x and $\varphi_a(x)$.

Lemma 5.3 *For any point $x \in \widehat{A}(s, 3\rho r_a)$, $\|\varphi_a(x)\, x\| \leqslant \pi \|s v_{a-1}\|_{\mathcal{T}}/2$.*

Proof. For any two points p and q on C_x, we use $C_x[p, q]$ to denote the shorter arc between p and q on C_x. It suffices to show that $\text{arclength}(C_x[x, \varphi_a(x)]) \leqslant \pi \|s v_{a-1}\|_{\mathcal{T}}/2$. Observe that for any point x on the arc $C'_x[y_i, y_{i+1}]$, $\text{arclength}(C_x[x, \varphi_a(x)])$ is the absolute value of an affine function of $\text{arclength}(C'_x[y_i, x])$. Hence, $\text{arclength}(C_x[x, \varphi_a(x)])$ achieves its maximum at the boundary of its domain, which means that $x = y_i$ and $\varphi_a(x) = z_i$, or $x = y_{i+1}$ and $\varphi_a(x) = z_{i+1}$. So it suffices to prove that $\text{arclength}(C_x[y_i, z_i])$ and $\text{arclength}(C_x[y_{i+1}, z_{i+1}])$ are at most $\pi \|s v_{a-1}\|_{\mathcal{T}}/2$.

Recall that e_i has an endpoint inside $B(s, 6\rho^2 r_{a-1})$ and that s is an endpoint of ℓ_i. Let w_i denote the endpoint of e_i inside $B(s, 6\rho^2 r_{a-1})$. Let h be the Euclidean distance from s to the supporting line of e_i. Refer to Figure 5. Both $\|w_i s\|$ and h are at most $6\rho^2 r_{a-1}$.

We bound $\text{arclength}(C_x[y_i, z_i])$ first. Let $\alpha = \angle s y_i w_i$ and $\beta = \angle y_i s z_i$. We can assume that $y_i \neq z_i$; otherwise, there is nothing to prove. So α and β belong to the range $(0, \pi/2]$ and $\alpha > \beta$. We have

$$\text{arclength}(C_x[y_i, z_i]) = \|s y_i\|\beta = h\beta/\sin\alpha < h\beta/\sin\beta.$$

The function $\beta/\sin\beta$ is maximized at $\beta = \pi/2$. So

$$\text{arclength}(C_x[y_i, z_i]) < h\pi/2 \leqslant 3\pi\rho^2 r_{a-1} = \pi \|s v_{a-1}\|_{\mathcal{T}}/2.$$

We can similarly bound $\text{arclength}(C_x[y_{i+1}, z_{i+1}])$ by $\pi \|s v_{a-1}\|_{\mathcal{T}}/2$. ◻

5.3 Data structures

Let a be an integer in the range $[2, n-1]$ such that $R_{a-1}/(2\rho) < r_a$. To handle query destinations that fall inside $\mathsf{Gap}(a)$, we build the data structure $\mathsf{CloseRange}(\mathcal{S}_a, \varepsilon/6)$ defined in Lemma 4.1. Given a query destination $t \in \mathsf{Gap}(a)$, we query $\mathsf{CloseRange}(\mathcal{S}_a, \varepsilon/6)$ with $\varphi_a(t)$, and transform the query results for $\varphi_a(t)$ back to the query results for t. All we have to do is to describe the transformation and prove its validity.

There are two technical issues in using $\mathsf{CloseRange}(\mathcal{S}_a, \varepsilon/6)$. First, some face in $\mathcal{S}_a$ may have angle larger than or equal to π. In this case, we split the face by inserting the angle bisector. Second, s is the only vertex in $\mathcal{S}_a$ and so ℓ_s, the distance from s to the nearest vertex, is not defined in $\mathcal{S}_a$. Recall that in Section 4, the motivation for using ℓ_s is to ensure that, whenever a query destination t is at distance at most $\ell_s/(5\rho)$ from s, any $(1+\varepsilon)$-approximate shortest path from s to t lies within the union of faces in $\mathcal{T}$ incident to s. In the case of $\mathcal{S}_a$, this is guaranteed since all faces are incident to s. Therefore, we can choose any value for r and an appropriate value for R correspondingly.

Specifically, we construct the data structure $\mathsf{PathQuery}(\mathcal{S}_a, 6\rho^2 r_{a-1}, 12\rho^3 r_{a-1}, \varepsilon/6)$. Given a query destination $t \in \mathsf{Gap}(a)$, we map t to $\varphi_a(t)$ and then proceed as in Section 4. We first compute the intersection point t'' between the boundary of $B(s, 6\rho^2 r_{a-1})$ and the ray from s through $\varphi_a(t)$. Since t'' is at distance $6\rho^2 r_{a-1}$ from s, the $(1+\varepsilon/6)$-approximate shortest path from s to t'' in $\mathcal{S}_a$ lies inside $B(s, 12\rho^3 r_{a-1})$ (by Lemma 2.1(ii)). Thus, the data structure $\mathsf{PathQuery}(\mathcal{S}_a, 6\rho^2 r_{a-1}, 12\rho^3 r_{a-1}, \varepsilon/6)$ reports a $(1+\varepsilon/6)$-approximate shortest path from s to t'' in $\mathcal{S}_a$. Then the approximate shortest path from s to t'' is scaled to a $(1+\varepsilon/6)$-approximate shortest path from s to $\varphi_a(t)$ in $\mathcal{S}_a$. It remains to transform the query results for $\varphi_a(t)$ back to the query results for t. Denote by $\mathrm{cost}_{\mathcal{S}_a}(P)$ the cost of a path P in $\mathcal{S}_a$. The next lemma relates the costs of the approximate shortest paths for t and $\varphi_a(t)$.

Lemma 5.4 *Assume that $R_{a-1}/(2\rho) < \|st\|_{\mathcal{T}} < r_a$; in other words, $t \in \mathsf{Gap}(a)$. Let P be a $(1+\varepsilon/6)$-approximate shortest path from s to $\varphi_a(t)$ in $\mathcal{S}_a$. Then $\mathrm{cost}_{\mathcal{S}_a}(P) \leqslant (1+\varepsilon/2)\mathsf{OPT}_{st}$.*

Proof. Let $k = 21\rho n^2/(\varepsilon/6)$. By Lemma 2.1(i), there is a k-link path Q_0 which is a $(1+\varepsilon/6)$-approximate shortest path from s to t in $|\mathcal{T}|$. Since $\|st\|_{\mathcal{T}} < r_a$, the path Q_0 lies inside $\widehat{B}(s, 2\rho r_a)$ by Lemma 2.1(ii).

Because $t \in \mathsf{Gap}(a)$ and $\mathsf{Gap}(a) \subseteq \widehat{A}(s, 3\rho r_a)$ by Lemma 5.2, the path Q_0 must enter $\widehat{A}(s, 3\rho r_a)$ at least once. We cut Q_0 at its last entry point q into $\widehat{A}(s, 3\rho r_a)$ to obtain a subpath Q_1 inside $\widehat{A}(s, 3\rho r_a)$. As Q_0 lies inside $\widehat{B}(s, 2\rho r_a)$, we know that q cannot be on the boundary of $B(s, 3\rho r_a)$, so it must lie on the boundary of $B(s, 6\rho^2 r_{a-1})$.

We make q a node of Q_1, if necessary. We apply the perturbation φ_a to the nodes of Q_1. For convenience, denote the path connecting the perturbed nodes by $\varphi_a(Q_1)$. We augment $\varphi_a(Q_1)$ by the segment $\overline{s\,\varphi_a(q)}$. This gives us a path from s to $\varphi_a(t)$ in $\mathcal{S}_a$. We now compare the cost of P with the cost of this new path. Since P is a $(1+\varepsilon/6)$-approximate shortest path, we get

$$\mathrm{cost}_{\mathcal{S}_a}(P) \leqslant (1+\varepsilon/6)(\mathrm{cost}_{\mathcal{S}_a}(\varphi_a(Q_1)) + \rho\|s\,\varphi_a(q)\|). \tag{1}$$

The number of nodes in Q_1 is at most $k+1$ (due to the possible insertion of q as a node). By the triangle inequality and Lemma 5.3, we get $\mathrm{cost}_{\mathcal{S}_a}(\varphi_a(Q_1)) \leqslant \mathrm{cost}(Q_0) + (k+1)\pi\rho\|sv_{a-1}\|_{\mathcal{T}}$. Also, $\rho\|s\,\varphi(q)\| = 6\rho^3 r_{a-1} = \rho\|sv_{a-1}\|_{\mathcal{T}}$. Plugging these into (1), we get

$$\mathrm{cost}_{\mathcal{S}_a}(P) \leqslant (1+\varepsilon/6)(\mathrm{cost}(Q_0) + (\pi k + \pi + 1)\rho\|sv_{a-1}\|_{\mathcal{T}}). \tag{2}$$

Choosing the parameter C_R in the definition of R_{a-1} to be at least $\pi \times 24 \times 128$, we have $(\pi k + \pi + 1)\rho\|sv_{a-1}\|_{\mathcal{T}} \leqslant \varepsilon R_{a-1}/(24\rho)$. Since we assume that $R_{a-1}/(2\rho) < \|st\|_{\mathcal{T}}$, we obtain $(\pi k + \pi +$

$1)\rho\|sv_{a-1}\|_{\mathcal{T}} < \frac{\varepsilon}{12}\|st\|_{\mathcal{T}} \leqslant \frac{\varepsilon}{12}\mathsf{OPT}_{st}$. We also have $\mathrm{cost}(Q_0) \leqslant \left(1+\frac{\varepsilon}{6}\right)\mathsf{OPT}_{st}$ by the definition of Q_0. Plugging these two inequalities into (2), we have $\mathrm{cost}_{\mathcal{S}_a}(P) \leqslant \left(1+\frac{\varepsilon}{6}\right)\left(1+\frac{\varepsilon}{6}+\frac{\varepsilon}{12}\right)\mathsf{OPT}_{st} \leqslant \left(1+\frac{\varepsilon}{2}\right)\mathsf{OPT}_{st}$, completing the proof. □

By Lemma 5.4, the cost of a $(1+\varepsilon/6)$-approximate shortest path P from s to $\varphi_a(t)$ in $\mathcal{S}_a$ is within the desired approximation bound. A natural postprocessing is to deform the path by applying φ_a^{-1} to the nodes of P in $\widehat{A}(s, 3\rho r_a)$. Since φ is not defined inside $B(s, 6\rho^2 r_{a-1})$, we have to handle the portion of P inside $B(s, 6\rho^2 r_{a-1})$ separately.

We describe our conversion of P from s to $\varphi_a(t)$ in $\mathcal{S}_a$ to a path $\tilde{P}$ from s to t in $|\mathcal{T}|$. Since $t \in \mathsf{Gap}(a) \subseteq \widehat{A}(s, 3\rho r_a)$, we have $\|st\| > 6\rho^2 r_{a-1}$. Because t and $\varphi_a(t)$ are at the same Euclidean distance from s, the point $\varphi_a(t)$ lies outside $B(s, 6\rho^2 r_{a-1})$. Thus, P must leave $B(s, 6\rho^2 r_{a-1})$ at some point. Let b be the point at which P leaves $B(s, 6\rho^2 r_{a-1}) \cap |\mathcal{T}|$ for the last time. We insert b as a new node into P. Denote the resulting sequence of nodes in P by $(s = p_0, p_1, \ldots, p_l = b, \ldots, p_m = \varphi_a(t))$. Let X' be the geodesic path from s to $\varphi_a^{-1}(b)$ in $B(s, 6\rho^2 r_{a-1}) \cap |\mathcal{T}|$. We define another path $X'' = (\varphi_a^{-1}(b) = \varphi_a^{-1}(p_l), \varphi_a^{-1}(p_{l+1}), \ldots, \varphi_a^{-1}(p_m) = t)$. Let $\tilde{P}$ be the polygonal path obtained by concatenating X' and X'' in this order. Summarizing this, we convert P to $\tilde{P}$ in three steps. (See Figure 6 for an illustration.)

1. Traverse P in $O\left(\frac{\rho^2 n^2}{\varepsilon^2}\log\rho\right)$ time to find the point b, at which P leaves $B(s, 6\rho^2 r_{a-1}) \cap |\mathcal{T}|$ for the last time. The complexity of P follows from Lemma 4.1.
2. Construct the geodesic path X' from s to $\varphi^{-1}(b)$ in time $O(\log n)$ plus the complexity of X', by using the method of Hershberger and Suri [13]. The complexity of X' is $O(n)$, and the method of Hershberger and Suri requires $O(n \log n)$ preprocessing time and $O(n \log n)$ storage.
3. Concatenate X' and X'' to form $\tilde{P}$.

In the following lemma, we will show that $\mathrm{cost}(\tilde{P}) \leqslant \mathrm{cost}_{\mathcal{S}_a}(P) + (2n + m\pi)\rho\|sv_{a-1}\|_{\mathcal{T}}$ and $\mathrm{cost}_{\mathcal{S}_a}(P) + (2n + m\pi)\rho\|sv_{a-1}\|_{\mathcal{T}} \leqslant (1+\varepsilon)\mathsf{OPT}_{st}$, where m is the number of nodes in P. It is easy to extend the data structure $\mathsf{CloseRange}(\mathcal{S}_a, \varepsilon/6)$ so that it reports the number m of nodes in P in $O(1)$ time after reporting $\mathrm{cost}_{\mathcal{S}_a}(P)$. So, by Lemma 4.1, a $(1+\varepsilon)$-approximation of OPT_{st} can be reported in $O(\log \frac{\rho n}{\varepsilon})$ time.

Lemma 5.5 *Assume that $R_{a-1}/(2\rho) < \|st\|_{\mathcal{T}} < r_a$. Let P be a $(1+\varepsilon/6)$-approximate shortest polygonal path from s to $\varphi_a(t)$ in $\mathcal{S}_a$. Let m be the number of nodes in P. Then* $\mathrm{cost}(\tilde{P}) \leqslant \mathrm{cost}_{\mathcal{S}_a}(P) + (2n + m\pi)\rho\,\|sv_{a-1}\|_{\mathcal{T}} \leqslant (1+\varepsilon)\mathsf{OPT}_{st}$.

Proof. Since X' is a geodesic path in $|\mathcal{T}|$ from s to $\varphi_a^{-1}(b)$, it consists of at most n segments lying in $B(s, 6\rho^2 r_{a-1})$. Thus each segment of X' has length at most $12\rho^2 r_{a-1}$, implying that $\mathrm{cost}(X') \leqslant n\rho \cdot 12\rho^2 r_{a-1} = 2n\rho\|sv_{a-1}\|_{\mathcal{T}}$. Using the triangle inequality and Lemma 5.3, we get

$$\mathrm{cost}(X'') \leqslant \mathrm{cost}_{\mathcal{S}_a}(P) + \sum_{i=l}^{m} 2\rho\,\|\varphi_a(p_i)\,p_i\| \leqslant \mathrm{cost}_{\mathcal{S}_a}(P) + m\pi\rho\,\|sv_{a-1}\|_{\mathcal{T}}.$$

Hence $\mathrm{cost}(\tilde{P}) \leqslant \mathrm{cost}_{\mathcal{S}_a}(P) + (2n + m\pi)\rho\,\|sv_{a-1}\|_{\mathcal{T}}$.

Lemma 4.1 implies that $m \leqslant \frac{864\rho n^2}{\varepsilon/6}\left(\frac{320\rho}{\varepsilon/6} + 7\right)\lceil\log(2\rho)\rceil$. By setting the parameter C_R in the definition of R_{a-1} to be a large enough multiple of π, we obtain $(2n + m\pi)\rho\,\|sv_{a-1}\|_{\mathcal{T}} < \frac{\varepsilon}{4\rho}R_{a-1}$. As we assume that $R_{a-1}/(2\rho) < \|st\|_{\mathcal{T}}$, we get $(2n + m\pi)\rho\|sv_{a-1}\|_{\mathcal{T}} < \frac{\varepsilon}{2}\|st\|_{\mathcal{T}} \leqslant$

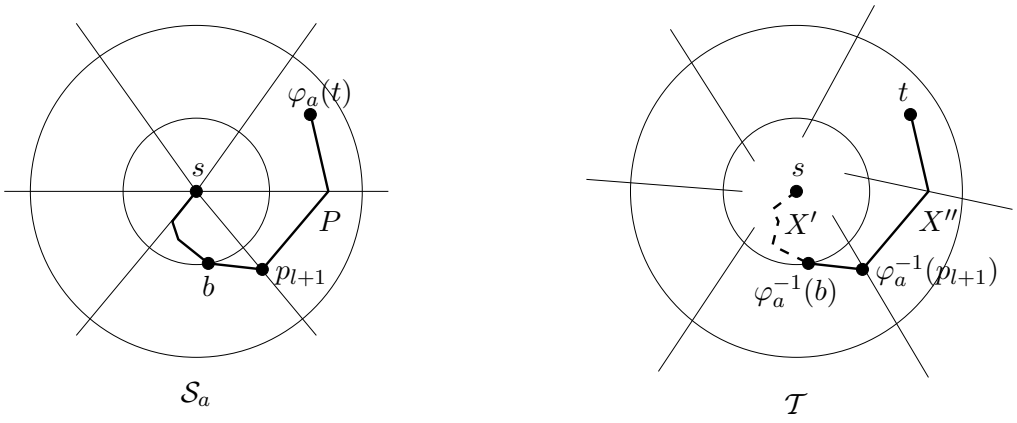

Figure 6: P is an approximate shortest polygonal path from s to $\varphi_a(t)$ in $\mathcal{S}_a$. We identify the point b at which P leaves $B(s, 6\rho^2 r_{a-1}) \cap |\mathcal{T}|$ for the last time. X' is the geodesic path from s to $\varphi_a^{-1}(b)$ and X'' is obtained by applying φ_a^{-1} to the nodes of the subpath of P from b to $\varphi_a(t)$. The path we want is $\tilde{P}$, the concatenation of X' and X''.

$\frac{\varepsilon}{2}\mathsf{OPT}_{st}$. Combining this result with Lemma 5.4 completes the proof. □

The following lemma summarizes our results in this section.

Lemma 5.6 *For any integer $a \in [2, n-1]$ such that $R_{a-1}/(2\rho) < r_a$, and for any $\varepsilon \in (0,1)$, we can construct a data structure $\mathsf{GapQuery}(\mathcal{T}, a, \varepsilon)$ such that:*

(i) *For any point $t \in |\mathcal{T}|$ such that $R_{a-1}/(2\rho) < \|st\|_{\mathcal{T}} < r_a$,*

- *a $(1+\varepsilon)$-approximation of the shortest path cost from s to t can be reported in $O\left(\log \frac{\rho n}{\varepsilon}\right)$ time;*
- *afterwards, a $(1+\varepsilon)$-approximate shortest path can be output in time $O(\log n)$ plus its complexity, and the complexity of this path is $O(\frac{\rho^2 n^2}{\varepsilon^2} \log \rho)$.*

(ii) *It uses $O(\frac{\rho^2 n^2}{\varepsilon^2} \log \rho)$ space.*

(iii) *It can be constructed in $O(\frac{\rho^2 n^2}{\varepsilon^2} \log \rho \log \frac{\rho n}{\varepsilon})$ time.*

6 Combining the data structures

We combine the data structures discussed in previous sections to produce a complete data structure to answer approximate shortest path queries in $\mathcal{T}$. We first build a Euclidean shortest path tree with the root s, using the method of Hershberger and Suri [13] in $O(n \log n)$ time and $O(n \log n)$ space. It also gives us the geodesic distances from s to all the vertices of $\mathcal{T}$. Let $s = v_0, v_1, v_2, v_3, \ldots, v_{n-1}$ be the vertices of $\mathcal{T}$ sorted in order of non-decreasing geodesic distances from s.

Simple combination. We motivate our strategy by first describing a simple combination of the data structures that does not give the best performance.

1. Construct $\mathsf{CloseRange}(\mathcal{T}, \varepsilon)$ to handle query points $t \in |\mathcal{T}|$ such that $\|st\|_{\mathcal{T}} \leqslant \ell_s/(5\rho) = \|sv_1\|_{\mathcal{T}}/(5\rho)$.

2. For each integer $a \in [1, n-1]$, construct $\mathsf{PathQuery}(\mathcal{T}, r_a, R_a, \varepsilon)$ to handle query points $t \in |\mathcal{T}|$ such that $r_a \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{1}{2\rho} R_a$.

3. For each integer $a \in [2, n-1]$ satisfying $\frac{1}{2\rho} R_{a-1} < r_a$, construct $\mathsf{GapQuery}(\mathcal{T}, a, \varepsilon)$ to handle query points $t \in |\mathcal{T}|$ such that $\frac{1}{2\rho} R_{a-1} < \|st\|_{\mathcal{T}} < r_a$.

By Lemma 4.1, the data structure in (1) takes $O\left(\frac{\rho^2 n^2}{\varepsilon^2} \log \rho\right)$ space and can be constructed in $O\left(\frac{\rho^2 n^2}{\varepsilon^2} \log \rho \log \frac{\rho n}{\varepsilon}\right)$ time. By Lemma 3.3, if we bound $|V^{\omega}_{r_a, R_a}|$ by n, the data structure in (2), which consists of $n-1$ $\mathsf{PathQuery}$ data structures, takes $O\left(\frac{\rho^2 n^4}{\varepsilon^2} \log \frac{\rho n}{\varepsilon}\right)$ space and can be built in $O\left(\frac{\rho^2 n^4}{\varepsilon^2} (\log \frac{\rho n}{\varepsilon})^2\right)$ time. By Lemma 5.6, the data structure in (3) takes $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \rho\right)$ space and can be built in $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \rho \log \frac{\rho n}{\varepsilon}\right)$ time.

The data structure in (2) dominates the space and preprocessing time complexities. Its space and preprocessing time complexities are larger than those in (1) and (3) by roughly a factor of n. This is because we compute $\mathsf{PathQuery}(\mathcal{T}, r_a, R_a, \varepsilon)$ for all integers $a \in [1, n-1]$ and bound $|V^{\omega}_{r_a, R_a}|$ by n for each integer $a \in [1, n-1]$. In the sequel, we improve the data structure in (2) by a factor of n in space requirement and preprocessing time, by computing a different set of values $\tilde{r}_a$ and $\tilde{R}_a$.

Modified structure. As a short hand, we use $\mathrm{poly}(\varepsilon)$ to denote the term $C_R \cdot \frac{\rho^4 n^2}{\varepsilon^3} \log(2\rho)$, and thus we have $R_a = \mathrm{poly}(\varepsilon) \cdot \|sv_a\|_{\mathcal{T}}$ by the definition of R_a. We first inductively extract a subset U of the vertices of $\mathcal{T}$. Later, we only define $\tilde{r}_i$ and $\tilde{R}_i$ for each vertex u_i in the subset U. (In contrast, the r_a and R_a are defined for every vertex v_a of $\mathcal{T}$.)

Initially, we set $u_1 = v_1$ and initialize $U = \{u_1\}$. For $i \geqslant 2$, we find two adjacent vertices v_{j-1} and v_j in $\mathcal{T}$ in the sorted order of geodesic distances from s such that

$$\|sv_{j-1}\|_{\mathcal{T}} \leqslant \mathrm{poly}(\varepsilon) \|su_{i-1}\|_{\mathcal{T}} < \|sv_j\|_{\mathcal{T}},$$

set $u_i = v_j$, and add u_i to U. We stop expanding U at the ith iteration when we cannot find a pair of vertices v_{j-1} and v_j in $\mathcal{T}$ such that $\|sv_{j-1}\|_{\mathcal{T}} \leqslant \mathrm{poly}(\varepsilon) \|su_{i-1}\|_{\mathcal{T}} < \|sv_j\|_{\mathcal{T}}$. Afterwards, for each $u_i \in U$, we define

$$\begin{aligned} \tilde{r}_i &= \frac{1}{6\rho^2} \cdot \|su_i\|_{\mathcal{T}} \\ \tilde{R}_i &= (\mathrm{poly}(\varepsilon))^2 \cdot \|su_i\|_{\mathcal{T}}. \end{aligned}$$

For each vertex $u_i \in U$, we construct the data structure $\mathsf{PathQuery}(\mathcal{T}, \tilde{r}_i, \tilde{R}_i, \varepsilon)$ to handle query points $t \in |\mathcal{T}|$ such that $\tilde{r}_i \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{1}{2\rho} \tilde{R}_i$. The idea is that we use the set of structures

$$\{\mathsf{PathQuery}(\mathcal{T}, r_{n-1}, R_{n-1}, \varepsilon)\} \cup \{\mathsf{PathQuery}(\mathcal{T}, \tilde{r}_i, \tilde{R}_i, \varepsilon) : u_i \in U\}$$

to replace the structures $\{\mathsf{PathQuery}(\mathcal{T}, r_a, R_a, \varepsilon) : \text{integer } a \in [1, n-1]\}$.

The next lemma shows that if a query point t is not handled by $\{\mathsf{PathQuery}(\mathcal{T}, \tilde{r}_i, \tilde{R}_i, \varepsilon) : u_i \in U\}$, then t is handled by either $\mathsf{CloseRange}(\mathcal{T}, \varepsilon)$, or $\mathsf{GapQuery}(\mathcal{T}, r_a, R_a, \varepsilon)$ for some integer $a \in [2, n-1]$, or $\{\mathsf{PathQuery}(\mathcal{T}, r_{n-1}, R_{n-1}, \varepsilon)\}$.

Lemma 6.1 *Let t be a point in $|\mathcal{T}|$. If $\tilde{r}_i \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{1}{2\rho}\tilde{R}_i$ does not hold for any $u_i \in U$, then either $\|st\|_{\mathcal{T}} < \frac{1}{5\rho}\|sv_1\|_{\mathcal{T}}$, or $\frac{1}{2\rho}R_{a-1} < \|st\|_{\mathcal{T}} < r_a$ for some integer $a \in [2, n-1]$, or $r_{n-1} \leqslant \|st\|_{\mathcal{T}} \leqslant \frac{1}{2\rho}R_{n-1}$.*

Proof. If $\|st\|_{\mathcal{T}} < \tilde{r}_1 = \frac{1}{6\rho^2}\|su_1\|_{\mathcal{T}} = \frac{1}{6\rho^2}\|sv_1\|_{\mathcal{T}}$, then $\|st\|_{\mathcal{T}} \leqslant \frac{1}{5\rho}\|sv_1\|_{\mathcal{T}}$. Otherwise, there are two cases:

- Case 1: $\frac{1}{2\rho}\tilde{R}_{i-1} < \|st\|_{\mathcal{T}} < \tilde{r}_i$ for some $u_i \in U$ and $i \geqslant 2$.
- Case 2: $\frac{1}{2\rho}\tilde{R}_b < \|st\|_{\mathcal{T}}$ where b is the largest vertex index in U.

Consider case 1. By definition, $u_i = v_a$ for some $2 \leqslant a \leqslant n$. Our inductive definition of u_i implies that $\|sv_{a-1}\|_{\mathcal{T}} \leqslant \text{poly}(\varepsilon)\|su_{i-1}\|_{\mathcal{T}} < \|sv_a\|_{\mathcal{T}}$. Thus, $R_{a-1} = \text{poly}(\varepsilon)\|sv_{a-1}\|_{\mathcal{T}} < (\text{poly}(\varepsilon))^2\|su_{i-1}\|_{\mathcal{T}} = \tilde{R}_{i-1}$ and $r_a = \frac{1}{6\rho^2}\|su_i\|_{\mathcal{T}} = \frac{1}{6\rho^2}\|sv_a\|_{\mathcal{T}} = \tilde{r}_i$. It follows that $\frac{1}{2\rho}R_{a-1} < \|st\|_{\mathcal{T}} < r_a$.

Consider case 2. Since b is the largest vertex index in U, it must be the case that $\text{poly}(\varepsilon)\|su_b\|_{\mathcal{T}} \geqslant \|sv_{n-1}\|_{\mathcal{T}}$. Thus, $\|st\|_{\mathcal{T}} > \frac{1}{2\rho}\tilde{R}_b = \frac{1}{2\rho}(\text{poly}(\varepsilon))^2\|su_b\|_{\mathcal{T}} \geqslant \frac{1}{2\rho}\text{poly}(\varepsilon)\|sv_{n-1}\|_{\mathcal{T}} > 3\|sv_{n-1}\|_{\mathcal{T}}$. On the other hand, t is in a triangle in $|\mathcal{T}|$. Since the vertex furthest away from s is v_{n-1}, by triangle inequality, any line segment inside the triangle containing t has length at most $2\|sv_{n-1}\|_{\mathcal{T}}$. But then, let v be a vertex of the triangle containing t, we have $\|st\|_{\mathcal{T}} \leqslant \|sv\|_{\mathcal{T}} + \|vt\| \leqslant 3\|sv_{n-1}\|_{\mathcal{T}}$, a contradiction. ◻

We are only left with the task of analyzing the space and preprocessing complexities of the data structure $\{\mathsf{PathQuery}(\mathcal{T}, \tilde{r}_i, \tilde{R}_i, \varepsilon) : u_i \in U\}$. By Lemma 3.3, it reduces to bounding the sum $\sum_{u_i \in U} \left|V^{\omega}_{\tilde{r}_i,\tilde{R}_i}\right|$ where $\omega = \varepsilon/8$. Recall that by Definition 2, we have

$$V^{\omega}_{\tilde{r}_i,\tilde{R}_i} \quad = \quad \left\{\text{vertex } v \text{ of } \mathcal{T} \,:\, \frac{\omega}{8\rho n}\tilde{r}_i < \|sv\|_{\mathcal{T}} < \frac{5}{2}\tilde{R}_i\right\}.$$

The next lemma shows that $V^{\omega}_{\tilde{r}_i,\tilde{R}_i}$ and $V^{\omega}_{\tilde{r}_j,\tilde{R}_j}$ are disjoint if i and j differ by more than 2.

Lemma 6.2 *For any $u_i, u_j \in U$, if $j > i+2$, then $V^{\omega}_{\tilde{r}_i,\tilde{R}_i} \cap V^{\omega}_{\tilde{r}_j,\tilde{R}_j}$ is empty.*

Proof. By the construction of U, we have

$$\|su_j\|_{\mathcal{T}} > \text{poly}(\varepsilon)\|su_{j-1}\|_{\mathcal{T}} > \cdots > (\text{poly}(\varepsilon))^{j-i}\|su_i\|_{\mathcal{T}} \geqslant (\text{poly}(\varepsilon))^3\|su_i\|_{\mathcal{T}}.$$

It follows that

$$\frac{\omega}{8\rho n}\tilde{r}_j = \frac{\omega}{48\rho^3 n}\|su_j\|_{\mathcal{T}} > \frac{\omega}{48\rho^3 n}(\text{poly}(\varepsilon))^3\|su_i\|_{\mathcal{T}} = \frac{\omega}{48\rho^3 n}\text{poly}(\varepsilon)\tilde{R}_i.$$

Substituting $\text{poly}(\varepsilon) = C_R \cdot \frac{\rho^4 n^2}{\varepsilon^3}\log(2\rho)$ and $\omega = \varepsilon/8$, and choosing C_R to be at least 48×20, we get

$$\frac{\omega}{8\rho n}\tilde{r}_j > \frac{C_R}{48} \cdot \frac{\omega}{\varepsilon^3}\rho n \cdot \log(2\rho) \cdot \tilde{R}_i \geqslant \frac{5}{2}\frac{\rho n}{\varepsilon^2}\tilde{R}_i \geqslant \frac{5}{2}\tilde{R}_i.$$

Hence, $V^{\omega}_{\tilde{r}_i,\tilde{R}_i} \cap V^{\omega}_{\tilde{r}_j,\tilde{R}_j}$ is empty. ◻

By Lemma 6.2, each vertex in $\mathcal{T}$ can appear in at most three consecutive $V^{\omega}_{\tilde{r}_i,\tilde{R}_i}$, $V^{\omega}_{\tilde{r}_{i+1},\tilde{R}_{i+1}}$, and $V^{\omega}_{\tilde{r}_{i+2},\tilde{R}_{i+2}}$. This implies the following result.

Corollary 2 *Any vertex v of $\mathcal{T}$ belongs to at most three sets in the collection $\{V^{\omega}_{\tilde{r}_i,\tilde{R}_i} : u_i \in U\}$. Thus $\sum_{u_i \in U} \left|V^{\omega}_{\tilde{r}_i,\tilde{R}_i}\right| \leqslant 3n$.*

Thus, the set of structures $\{\mathsf{PathQuery}(\mathcal{T}, r_{n-1}, R_{n-1}, \varepsilon)\} \cup \{\mathsf{PathQuery}(\mathcal{T}, \tilde{r}_i, \tilde{R}_i, \varepsilon) : u_i \in U\}$ requires $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon}\right)$ space and can be built in $O\left(\frac{\rho^2 n^3}{\varepsilon^2} (\log \frac{\rho n}{\varepsilon})^2\right)$ time. Combining this set with the structures $\mathsf{CloseRange}(\mathcal{T}, \varepsilon)$ and $\mathsf{GapQuery}(\mathcal{T}, a, \varepsilon)$ for every integer $a \in [2, n-1]$ such that $\frac{1}{2\rho} R_{a-1} < r_a$, we obtain the main theorem of this paper.

Theorem 1 *Let $\mathcal{T}$ be a planar subdivision such that each face is associated with a convex distance function. Let s be a fixed vertex in $\mathcal{T}$. For any $\varepsilon \in (0, 1)$, we can construct a data structure such that:*

(i) *For any point $t \in |\mathcal{T}|$,*

- *a $(1+\varepsilon)$ approximation of the shortest path cost from s to t can be reported in $O\left(\log \frac{\rho n}{\varepsilon}\right)$ time;*
- *afterwards, a $(1+\varepsilon)$-approximate shortest path can be output in time $O(\log n)$ plus its complexity, and the complexity of this path is $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon}\right)$.*

(ii) *It uses $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \log \frac{\rho n}{\varepsilon}\right)$ space.*

(iii) *It can be built in $O\left(\frac{\rho^2 n^3}{\varepsilon^2} \left(\log \frac{\rho n}{\varepsilon}\right)^2\right)$ time.*

7 Conclusion

We developed a data structure to answer approximate shortest path queries from a fixed source in a planar subdivision. Our cost model generalizes previous non-Euclidean cost models. Most notably, the performance of our data structure is independent of the geometric parameters of the environment such as the minimum angle in the subdivision. When specializing to the weighted region model, ρ is the ratio of the maximum weight to the minimum weight, and our dependence on ρ is worse than the logarithmic dependence on ρ of Mitchell and Papadimitriou's algorithm [17]. There are two natural directions for future research. First, can the dependence on ρ and n be lowered? This is related to the problem of designing a faster single-source approximate shortest path algorithm. Second, can two-point queries be supported? The scaling transformation and the perturbation technique introduced in this paper are not powerful enough to handle an arbitrary source vertex.

8 Acknowledgment

We thank the referees for their helpful suggestions.

References

[1] H.-K. Ahn, P. Brass, O. Cheong, H.-S. Na, C.-S. Shin, and A. Vigneron. Inscribing an axially symmetric polygon and other approximation algorithms for planar convex sets. *Computational Geometry: Theory and Applications*, 33 (2006), 152–164.

[2] L. Aleksandrov, H. Djidjev, H. Guo, A. Maheshwari, D. Nussbaum, and J.-R. Sack. Approximate shortest path queries on weighted polyhedral surfaces. *Proceedings of the 31st International Symposium on Mathematical Foundations of Computer Science*, 2006, 98–109.

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

[4] S. Arya and A. Vigneron. Approximating a Voronoi cell. Technical Report HKUST-TCSC-2003-10, Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, 2003.

[5] S.-W. Cheng, H.-S. Na, A. Vigneron, and Y. Wang. Querying approximate shortest paths in anisotropic regions. *Proceedings of the 23rd Annual Symposium on Computational Geometry*, 2007, 84–91.

[6] S.-W. Cheng, H.-S. Na, A. Vigneron, and Y. Wang. Approximate shortest paths in anisotropic regions. *SIAM Journal on Computing*, 38 (2008), 802–824.

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

[8] Y.-J. Chiang and J. Mitchell. Two-point Euclidean shortest path queries in the plane. *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1999, 215–224.

[9] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2008.

[10] R. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *Journal of Approximation Theory*, 10 (1974), 227–236.

[11] S. Har-Peled. Approximate shortest paths and geodesic diameter on a convex polytope in three dimensions. *Discrete and Computational Geometry*, 21 (1999), 217–231.

[12] S. Har-Peled. Constructing approximate shortest path maps in three dimensions. *SIAM Journal on Computing*, 28 (1999), 1182–1197.

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

[14] M. Lanthier, A. Maheshwari, and J.-R. Sack. Approximating shortest paths on weighted polyhedral surfaces. *Algorithmica*, 30 (2001), 527–562.

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

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

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

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

[19] Y. Schreiber and M. Sharir. An optimal-time algorithm for shortest paths on a convex polytope in three dimensions. *Proceedings of the 22nd Annual Symposium on Computational Geometry*, 2006, 30–39.

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

A Refined analysis

In our previous paper [6], we proved that there exists a $(1+\varepsilon)$-approximate shortest polygonal path from s to t with $O(\rho n^2/\varepsilon)$ links. We are to refine this bound. The analysis in reference [6] takes any polygonal path P from s to t and shows how to convert it to another path with $O(\rho n^2/\varepsilon)$ links such that the path cost increases by no more than $O(\varepsilon \text{ cost}(P))$. The conversion first makes P simple and removes redundant turns, and then snaps the path to nearby vertices of $\mathcal{T}$.

Our idea is to revise the final snapping step. Specifically, we only snap path nodes in the vicinity of s or in the vicinity of vertices of $\mathcal{T}$ that are within a range of geodesic distances from s. This allows us to control the path complexity while keeping the increase in path cost tolerable.

We need some notation about polygonal paths from reference [6]. A *$\mathcal{T}$-respecting path* is a polygonal path P such that each link is contained in a face of $\mathcal{T}$. For example, a k-link path from s to t is $\mathcal{T}$-respecting. On the other hand, a $\mathcal{T}$-respecting path may have a node in the interior of a face but a k-link path cannot. Given a polygonal path $P = (p_0, p_1, p_2, \ldots)$, we use $P[i, j]$ to denote the subpath $(p_i, p_{i+1}, \ldots, p_j)$. Given two polygonal paths P and Q, we use $P \cdot Q$ to denote their concatenation in this order.

There is a technical issue that the analysis in reference [6] assumes that t is a vertex of $\mathcal{T}$. We enforce this by inserting t and splitting the triangular face containing t into three smaller triangles. For convenience, we still use $\mathcal{T}$ to denote the modified subdivision, but the number of vertices increases to $n+1$. We use the "original $\mathcal{T}$" when we need to refer to the subdivision before adding t. We will discuss how to revert to the original $\mathcal{T}$ at the end of this section.

In this section, we assume that $n \geqslant 7(|V_{st}^{\varepsilon}| + 2)$. Otherwise, Lemma 2.2 is directly implied by Lemma 2.1(i).

A.1 Zone

We define the "vicinity" of a vertex in $\mathcal{T}$. Let r be a real number such that $r \geqslant 0$. For any vertex v of $\mathcal{T}$, denote by $\widehat{B}(v, r)$ the connected component of $B(v, r) \cap |\mathcal{T}|$ that contains v. Let $T_{v,r}$ be the set of faces with a vertex in $\widehat{B}(v, r)$. For each face $f \in T_{v,r}$, we call a vertex w of f an *outside vertex* if w lies outside $\widehat{B}(v, r)$, and an edge of f an *outside edge* if it connects two outside vertices. For each face $f \in T_{v,r}$, remove its outside vertices and outside edges from f and denote the resulting face by $f_{v,r}$. Either $f_{v,r}$ is equal to f, or $f_{v,r}$ is a semi-open set. Figure 7 gives an illustration.

Definition 8 *For any real numbers δ and r such that $\delta > r \geqslant 0$, and for any vertex v of $\mathcal{T}$, the* (δ, r)-zone *of v is $B(v, \delta) \cap \bigcup_{f \in T_{v,r}} f_{v,r}$.*

Figure 8 gives an example of a (δ, r)-zone. Making the faces possibly semi-open provides the following nice property.

Lemma A.1 *Let $P = (p_0, p_1, p_2, \ldots)$ be a $\mathcal{T}$-respecting path. Let u be a vertex of $\mathcal{T}$. If a node p_i of P is in the (δ, r)-zone of u for some $\delta > r \geqslant 0$, then there exist a face in $T_{u,r}$ that contain $\overline{p_i p_{i+1}}$ and a face in $T_{u,r}$ that contain $\overline{p_{i-1} p_i}$.*

Proof. We prove the lemma for $\overline{p_i p_{i+1}}$. The same proof works for $\overline{p_{i-1} p_i}$. Suppose that p_i does not lie on any edge of $\mathcal{T}$. The node p_i is in the (δ, r)-zone of u, so there is a unique face $f \in T_{u,r}$

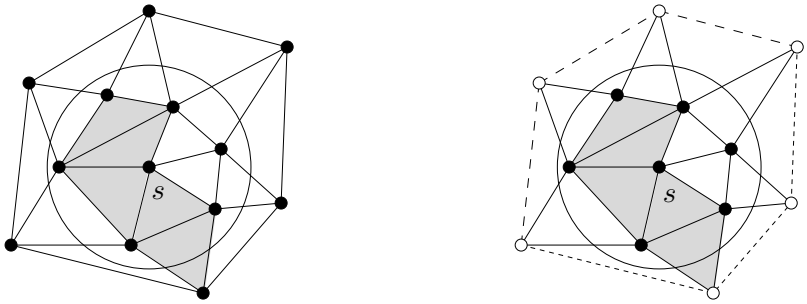

Figure 7: In each figure, the circle denotes $B(s, r)$, the shaded regions are obstacles. The faces f in $T_{v,r}$ are shown on the left, and the faces $f_{v,r}$ with the outside vertices and outside edges deleted are shown on the right. White dots and dashed edges indicate outside vertices and outside edges, respectively.

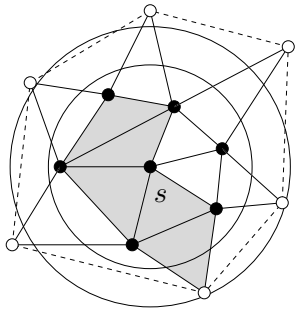

Figure 8: The larger circle and smaller circle denote $B(s, \delta)$ and $B(s, r)$, respectively. The (δ, r)-zone of s consists of the points in the unshaded triangles inside the larger circle, except those points that lie at the white dots or on the dashed edges.

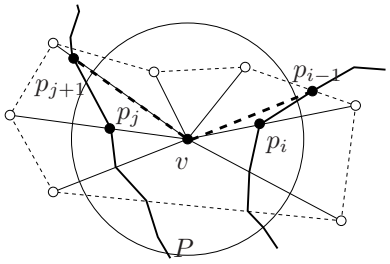

Figure 9: The $(\delta, 0)$-zone of v consists of the portions of the triangles inside the circle other than the white dots and the dashed edges. The bold polylines are portions of the path P, and the dashed polyline is the shortcut via v.

such that $p_i \in \text{int}(f)$. As P is $\mathcal{T}$-respecting, p_{i+1} must lie inside f or on its boundary, i.e., f contains $\overline{p_i p_{i+1}}$.

Suppose that p_i belongs to the interior of an edge e of $\mathcal{T}$. Then e is not an outside edge, implying that at least one endpoint of e lies in $\widehat{B}(u, r)$. This means that every face incident to e belongs to $T_{u,r}$. If p_{i+1} lies on e, then e contains $\overline{p_i p_{i+1}}$ and so does each face incident to e. If p_{i+1} does not lie on e, then p_{i+1} must belong to a face incident to e, as P is $\mathcal{T}$-respecting. So this face contains $\overline{p_i p_{i+1}}$.

Suppose that p_i is a vertex of $\mathcal{T}$. The vertex p_i must belong to $\widehat{B}(u, r)$ in order to be in the (δ, r)-zone of u. So all faces incident to p_i belong to $T_{u,r}$. The same argument shows that one of these faces contains $\overline{p_i p_{i+1}}$. □

A.2 Shortcut and Bypass

We use two procedures, Shortcut and Bypass, to snap path nodes. Shortcut was introduced in reference [6] and we use it to snap path nodes near some vertices of $\mathcal{T}$ other than the source s. We introduce a new procedure Bypass to snap path nodes near s.

Given a $\mathcal{T}$-respecting path $P = (p_0, p_1, \ldots)$ and a vertex v of $\mathcal{T}$, the procedure Shortcut(P, δ, v) finds the first node p_i and the last node p_j of P inside the $(\delta, 0)$-zone of v.[2] (Notice that we have set r to be zero.) Then, Shortcut replaces $P[i-1, j+1]$ with the subpath (p_{i-1}, v, p_{j+1}). The following is the pseudo-code of Shortcut. (Recall that s is a vertex of $\mathcal{T}$ and t has also been inserted as a vertex.) The output path of Shortcut may be self-intersecting, but it is still $\mathcal{T}$-respecting. Figure 9 shows an example.

Shortcut(Polygonal path P, $\delta > 0$, vertex v)

1. If no node of P is in the $(\delta, 0)$-zone of v, return P.
2. Let p_i and p_j be the first and last nodes along P, respectively, that lie in the $(\delta, 0)$-zone 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 $P[0, i-1] \cdot (v) \cdot P[j+1, m]$.

[2]The $(\delta, 0)$-zone of v is called the δ-neighborhood of v in reference [6].

The following result about Shortcut was proved in reference [6]. It follows from the observation that the shortcut (p_{i-1}, v, p_{j+1}) incurs an extra cost of $\text{cost}(\overline{p_{i-1}v}) - \text{cost}(\overline{p_{i-1}p_i}) + \text{cost}(\overline{vp_{j+1}}) - \text{cost}(\overline{p_jp_{j+1}}) \leqslant d_{f_1}(p_i, v) + d_{f_2}(v, p_j) \leqslant \rho(\|p_iv\| + \|vp_j\|) < 2\rho\delta$, where f_1 and f_2 are the faces in $T_{v,0}$ that contain $\overline{p_{i-1}p_i}$ and $\overline{p_jp_{j+1}}$, respectively (Lemma A.1).

Lemma A.2 ([6]) *Let P be a $\mathcal{T}$-respecting path. Let v be a vertex of $\mathcal{T}$. The path returned by* Shortcut(P, δ, v) *has cost less than* $\text{cost}(P) + 2\rho\delta$.

Given a $\mathcal{T}$-respecting path $P = (s = p_0, p_1, \ldots, p_m = t)$, the procedure Bypass$(P, \delta, r)$ identifies the last node p_j of P inside the (δ, r)-zone of s. In general, by Lemma A.1, there is a vertex v inside $\widehat{B}(s, r)$ lying in the same face as p_j and p_{j+1}. Bypass computes the geodesic path Q in $\widehat{B}(s, r)$ from s to v, and splits Q at its intersections with the edges of $\mathcal{T}$ to make it $\mathcal{T}$-respecting. Denote the split path by Split(Q). Bypass then replaces $P[0, j]$ with Split$(Q) \cdot (v)$. In the special case of $p_j = p_m$, we replace $P[0, j-1]$ in a similar way instead. The output of Bypass may be self-intersecting, but it remains $\mathcal{T}$-respecting. The following is the pseudo-code of Bypass. Figure 10 shows an example.

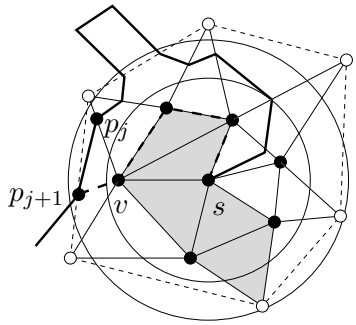

Figure 10: This is the (δ, r)-zone of s from Figure 8. The bold polyline denotes the prefix of the path P. The dashed polyline denotes the bypass from s to p_{j+1}.

Bypass($\mathcal{T}$-respecting path P from s to t, real numbers δ and r such that $\delta > r \geqslant 0$)

1. Let p_j be the last node along P that lies in the (δ, r)-zone of s. Let p_m be the last node of P.
2. If $p_j = p_m$, let $\ell = j - 1$; otherwise, let $\ell = j$.
3. Let $f \in T_{s,r}$ be a face containing $\overline{p_\ell p_{\ell+1}}$. Let v be a vertex of f in $\widehat{B}(s, r)$.
4. If $v = s$, return $(s) \cdot P[\ell + 1, m]$.
5. Compute the geodesic path Q in $\widehat{B}(s, r)$ from s to v.
6. Return Split$(Q) \cdot P[\ell + 1, m]$.

The next result gives some properties of the output path of Bypass.

Lemma A.3 *Let P be a $\mathcal{T}$-respecting path from s to t. Let R be the path returned by* Bypass(P, δ, r) *for some $\delta > r \geqslant 0$. Let v be the vertex picked by* Bypass *in step 3.*

(i) *The subpath of R from s to v lies in $\widehat{B}(s, r)$ and has at most $4n - 2$ nodes.*

(ii) $\text{cost}(R) \leqslant \text{cost}(P) + 2\rho nr + 2\rho\delta$.

Proof. The subpath of R from s to v is obtained by splitting a geodesic path in $\widehat{B}(s,r)$ from s to v at the intersections with the edges of $\mathcal{T}$. So this subpath lies in $\widehat{B}(s,r)$. The geodesic path has at most $n+1$ nodes because each node including s and v must be a distinct vertex of $\mathcal{T}$. (Recall that $\mathcal{T}$ has $n+1$ vertices after the addition of t.) Since a geodesic path cannot cross the same edge of $\mathcal{T}$ twice, splitting this geodesic path at the edges of $\mathcal{T}$ produces at most $3(n+1)-6 = 3n-3$ extra nodes. Thus, the subpath of R from s to v has at most $4n-2$ nodes. This proves (i).

The cost of R is at most the cost of the subpath of R from s to v plus $\text{cost}(P)+\text{cost}(\overline{vp_{\ell+1}})-\text{cost}(\overline{p_\ell p_{\ell+1}}) \leqslant \text{cost}(P)+\text{cost}(\overline{vp_\ell})$. The link $\overline{vp_\ell}$ lies inside $B(s,\delta)$, so its length is at most 2δ. The subpath from s to v comes from a geodesic path with at most n segments, each with length at most $2r$. Thus, the total extra cost is at most $2\rho nr + 2\rho\delta$. This proves (ii). ◻

A.3 Modify

In this subsection, we describe the conversion of a $\mathcal{T}$-respecting 2-approximate shortest path P into a path with few links such that the increase in cost is relatively small. We show that this conversion guarantees some structural properties and a bound on the cost of the output path. In turn, these results allow us to prove Lemma 2.2 in Section A.4. We first need some definitions and a background result.

Definition 9 *For any point $t \in |\mathcal{T}|$, define*

$$\begin{aligned} \gamma^{\varepsilon}_{st} &= \frac{\varepsilon}{8\rho n}\|st\|_{\mathcal{T}} \\ \delta^{\varepsilon}_{st} &= \frac{\varepsilon}{8\rho(|V^{\varepsilon}_{st}|+2)}\mathsf{OPT}_{st} \end{aligned}$$

Notice that $\gamma^{\varepsilon}_{st} \leqslant \delta^{\varepsilon}_{st}/7$ for any $\varepsilon \in (0,1)$ based on our assumption that $n \geqslant 7(|V^{\varepsilon}_{st}|+2)$. Using $\gamma^{\varepsilon}_{st}$, we can rewrite the definition of V^{ε}_{st} as

$$V^{\varepsilon}_{st} = \{\text{vertex } v \text{ of the original } \mathcal{T} : \gamma^{\varepsilon}_{st} < \|sv\|_{\mathcal{T}} < 5\rho\,\|st\|_{\mathcal{T}}\}.$$

A node on a $\mathcal{T}$-respecting path is called *critical* if (i) it is different from any vertex of $\mathcal{T}$ (including s and t), (ii) it is incident to two links, and (iii) exactly one of these two links is contained in an edge of $\mathcal{T}$. A node on a $\mathcal{T}$-respecting path is called *transversal* if (i) it lies in the interior of an edge e of $\mathcal{T}$, (ii) it is incident to two links, and (iii) the interior of the two links are in the interior of the two faces incident to e respectively. The following result is from reference [6].

Lemma A.4 ([6]) *Let P be a $\mathcal{T}$-respecting path from s to t. There is a procedure* $\mathsf{Simplify}(P)$ *that outputs a path Q with the following properties.*

(i) *Q is simple and $\mathcal{T}$-respecting. Each node of Q is either a vertex of $\mathcal{T}$, a critical node, or a transversal node.*

(ii) *Q has no more links than P.*

(iii) $\text{cost}(Q) \leqslant \text{cost}(P)$.

(iv) *Let u and v be two nodes of Q such that they are vertices of $\mathcal{T}$ and the nodes of Q between u and v are critical or transversal. There are at most two critical nodes of Q between u and v. Hence, Q has at most $2n$ critical nodes.*

Our conversion procedure Modify uses the procedures Simplify, Shortcut and Bypass. Its pseudo-code is given below.

Modify($\mathcal{T}$-respecting path P from s to t, $\varepsilon \in (0,1)$)

1. $Q_0 := \mathsf{Simplify}(P)$.
2. $Q_1 := \mathsf{Bypass}(Q_0, \delta^\varepsilon_{st}, \gamma^\varepsilon_{st})$.
3. Let $V^\varepsilon_{st} \cup \{t\} = \{v_1, v_2, v_3, \ldots\}$. For $i := 1$ to $|V^\varepsilon_{st}|+1$, $Q_{i+1} := \mathsf{Shortcut}(Q_i, \delta^\varepsilon_{st}, v_i)$.
4. Return $Q_{|V^\varepsilon_{st}|+2}$.

The next three lemmas give some structural properties and a bound on the cost of the output path of Modify.

Lemma A.5 *Let P be a $\mathcal{T}$-respecting 2-approximate shortest path from s to t. Let $R = (r_1, r_2, \ldots, r_\ell)$ be the path returned by* Modify(P, ε).

(i) *R is a $\mathcal{T}$-respecting path with distinct nodes. Each node of R lies on the boundary of a face of $\mathcal{T}$.*

(ii) $\mathrm{cost}(R) \leqslant \mathrm{cost}(P) + \dfrac{\varepsilon}{2}\mathsf{OPT}_{st}$.

(iii) *For any node r_i, if r_i is a vertex of $\mathcal{T}$ and $r_i \notin \widehat{B}(s, \gamma^\varepsilon_{st})$, then $r_i \in V^\varepsilon_{st} \cup \{t\}$.*

(iv) *If r_i and r_{i+1} are outside $\widehat{B}(s, \gamma^\varepsilon_{st})$ and neither of them is a vertex of $\mathcal{T}$, then $\overline{r_i r_{i+1}}$ is a link of Q_0.*

(v) *Let v be a vertex of $\mathcal{T}$. Let r_i be a node in the interior of an edge incident to v. If v lies inside $\widehat{B}(s, \gamma^\varepsilon_{st})$ and r_i is outside $\widehat{B}(s, \gamma^\varepsilon_{st})$, then $\|r_i s\| \geqslant \delta^\varepsilon_{st}$. If $v \in V^\varepsilon_{st} \cup \{t\}$, then $\|r_i v\| \geqslant \delta^\varepsilon_{st}$.*

(vi) *Among the set of nodes of R outside $\widehat{B}(s, \gamma^\varepsilon_{st})$, at most $|V^\varepsilon_{st}| + 1$ of them are vertices of $\mathcal{T}$ and at most $4|V^\varepsilon_{st}| + 5$ of them are critical nodes.*

Proof. By Lemma A.4(i), Q_0 is a $\mathcal{T}$-respecting path that satisfies the properties in (i). The calls to Bypass and Shortcut preserve these properties, thus (i) is proved.

By Lemma A.3, Bypass incurs an extra cost of at most

$$2\rho n \gamma^\varepsilon_{st} + 2\rho\delta^\varepsilon_{st} \leqslant \frac{\varepsilon\|st\|_{\mathcal{T}}}{4} + \frac{\varepsilon\mathsf{OPT}_{st}}{4|V^\varepsilon_{st}| + 8} \leqslant \frac{\varepsilon\mathsf{OPT}_{st}}{4} + \frac{\varepsilon\mathsf{OPT}_{st}}{4|V^\varepsilon_{st}| + 8}.$$

By Lemma A.2, the calls to Shortcut increase the path cost by at most

$$2\left(|V^\varepsilon_{st}| + 1\right)\rho\delta^\varepsilon_{st} = \frac{\varepsilon\left(|V^\varepsilon_{st}| + 1\right)\mathsf{OPT}_{st}}{4|V^\varepsilon_{st}| + 8}.$$

Hence, the total increase in path cost is at most $\varepsilon\mathsf{OPT}_{st}/2$, which proves (ii).

By (ii) and the assumption that $\mathrm{cost}(P) \leqslant 2\mathsf{OPT}_{st}$, we get $\mathrm{cost}(R) < 3\mathsf{OPT}_{st} < 5\rho\|st\|_{\mathcal{T}}$, which implies that $\|sr_i\|_{\mathcal{T}} < 5\rho\|st\|_{\mathcal{T}}$ for any node r_i of R. Thus, if r_i is a vertex of $\mathcal{T}$ and $r_i \notin \widehat{B}(s, \gamma^\varepsilon_{st})$, then $r_i \in V^\varepsilon_{st} \cup \{t\}$. This proves (iii).

Property (iv) follows directly from the working of Shortcut.

Consider (v). Suppose that v is a vertex of $\mathcal{T}$ in $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. Take an edge e of $\mathcal{T}$ incident to v and a node r_i of R in $\mathrm{int}(e)$ outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. Bypass only creates nodes inside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. Therefore, r_i is a node from Q_0. It survives the call of Bypass only if r_i lies outside $B(s, \delta^{\varepsilon}_{st})$. That is, $\|r_i s\| \geqslant \delta^{\varepsilon}_{st}$. Similarly, for any $v \in V^{\varepsilon}_{st} \cup \{t\}$ and for any r_i in the interior of an edge incident to v, the node r_i survives the call of Shortcut only if $\|r_i v\|_{\mathcal{T}} \geqslant \delta^{\varepsilon}_{st}$.

Consider (vi). Let r_i be a node of R outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. It follows from (iii) that if r_i is a vertex $\mathcal{T}$, then $r_i \in V^{\varepsilon}_{st} \cup \{t\}$. This proves the first part of (vi). Suppose that r_i is not a vertex of $\mathcal{T}$. Let Q'_1 be the suffix of Q_1 whose nodes are outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. So r_i is a node of Q'_1. By Lemmas A.3(ii) and A.4(iii) and our assumption on $\mathrm{cost}(P)$, we have $\mathrm{cost}(Q_1) \leqslant \mathrm{cost}(P) + 2\rho n \gamma^{\varepsilon}_{st} + 2\rho\delta^{\varepsilon}_{st} \leqslant 2\mathsf{OPT}_{st} + 2\rho n \gamma^{\varepsilon}_{st} + 2\rho\delta^{\varepsilon}_{st}$. Then, as in proving (iii), we can show that any vertex of $\mathcal{T}$ in Q'_1 belongs to $V^{\varepsilon}_{st} \cup \{t\}$. Since the nodes of Q'_1 are all inherited from Q_0, by Lemma A.4(iv), there are at most two critical nodes between any two successive nodes in Q'_1 that are vertices of $\mathcal{T}$. So the number of critical nodes in Q'_1 is at most $2(|V^{\varepsilon}_{st}|+2)-2 = 2|V^{\varepsilon}_{st}|+2$. (We add two to $|V^{\varepsilon}_{st}|$ to account for t and the vertex in Q_0 preceding Q'_1 .) Each call of Shortcut may introduce at most two new critical nodes, i.e., the endpoints of the subpath that it replaces. So the calls to Shortcut adds at most $2|V^{\varepsilon}_{st}| + 2$ critical nodes. Bypass may add at most one critical node outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. Therefore, in total, R has at most $4|V^{\varepsilon}_{st}| + 5$ critical nodes outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. □

The next lemma gives a lower bound on the cost of a subpath of the output path of Modify under certain conditions.

Lemma A.6 *Let P be a $\mathcal{T}$-respecting 2-approximate shortest path from s to t. Let R be the path returned by* Modify(P, ε). *Consider an edge e of $\mathcal{T}$ and a node r_i of R in* $\mathrm{int}(e)$. *Suppose that the following conditions hold:*

- *There exists a node r_j of R 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 outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$.*

Then $\mathrm{cost}(R[i,j]) > \delta^{\varepsilon}_{st}/2$.

Proof. The subpath $R[i,j]$ and the segment $\overline{r_i r_j}$ bound a closed region. We first prove that there is an edge e' of $\mathcal{T}$ that has an endpoint v' inside this closed region. Let E denote the set of edges of $\mathcal{T}$ that contain a node in the interior of $R[i,j]$. Pick an edge e'' of E. If this edge has an endpoint inside the closed region, we are done. Otherwise, this edge crosses $\mathrm{int}(R[i,j])$ transversally and so it separates a portion of the closed region from e. By recursively applying the argument on this portion of the closed region, we must find an edge e' of $\mathcal{T}$ that has an endpoint v' inside the closed region.

Let r_k be the node in $\mathrm{int}(R[i,j])$ that e' passes through. If $v' \in V^{\varepsilon}_{st} \cup \{t\}$, Lemma A.5(v) implies that $\|r_k v'\| \geqslant \delta^{\varepsilon}_{st}$. It follows that $\mathrm{cost}(R[i,j]) \geqslant \mathrm{length}(R[i,j]) \geqslant \|r_k v'\| \geqslant \delta^{\varepsilon}_{st}$.

Suppose that $v' \notin V^{\varepsilon}_{st} \cup \{t\}$. We first claim that $v' \in \widehat{B}(s, \gamma^{\varepsilon}_{st})$. Since $\|r_k v'\| \leqslant \mathrm{length}(R[i,j])$, we have

$$\|sv'\|_{\mathcal{T}} \leqslant \mathrm{length}(R[1,k]) + \|r_k v'\| \leqslant \mathrm{length}(R[1,k]) + \mathrm{length}(R[i,j]) \leqslant 2\ \mathrm{length}(R).$$

Since $\mathrm{cost}(R) \leqslant \mathrm{cost}(P) + \varepsilon\mathsf{OPT}_{st}/2$ by Lemma A.5(ii) and $\mathrm{cost}(P) \leqslant 2\mathsf{OPT}_{st}$ by assumption, we have $\mathrm{cost}(R) < \frac{5}{2}\mathsf{OPT}_{st}$. Thus,

$$\|sv'\|_{\mathcal{T}} \leqslant 2\ \mathrm{length}(R) \leqslant 2\ \mathrm{cost}(R) < 5\rho\|st\|_{\mathcal{T}}.$$

It follows that $\|sv'\|_{\mathcal{T}}$ satisfies the upper bound in the definition of V^{ε}_{st}. In order that $v' \notin V^{\varepsilon}_{st} \cup \{t\}$, we have $\|sv'\|_{\mathcal{T}} < \gamma^{\varepsilon}_{st}$ and so $v' \in \widehat{B}(s, \gamma^{\varepsilon}_{st})$. This proves our claim.

By our claim and Lemma A.5(v), $\|r_k s\| \geqslant \delta^{\varepsilon}_{st}$. So $\|r_k v'\| \geqslant \delta^{\varepsilon}_{st} - \gamma^{\varepsilon}_{st} > \delta^{\varepsilon}_{st}/2$. It follows that $\mathrm{cost}(R[i,j]) \geqslant \mathrm{length}(R[i,j]) > \|r_k v'\| > \delta^{\varepsilon}_{st}/2$. □

The next result bounds the complexity of the subpath of R consisting of the nodes outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$.

Lemma A.7 *Let P be a $\mathcal{T}$-respecting 2-approximate shortest path from s to t. Let R be the path returned by* $\mathsf{Modify}(P, \varepsilon)$. *Let R' be the longest suffix of R whose nodes are outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. The subpath R' has at most $3n(|V^{\varepsilon}_{st}| + 2)\,(40\rho/\varepsilon + 5) + 4n$ nodes.*

Proof. Let $Q_0 = \mathsf{Simplify}(P)$. For an edge $e \in \mathcal{T}$, we denote by N_e the set of nodes r_i of R' other than its two endpoints that lie in the interior of e. We bound the cardinality of N_e by a charging argument. 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 $r_i \in \mathrm{int}(e)$, r_{i-1} must be a vertex of the faces incident to e. There are 4 vertices in the two faces incident to e. It follows that case 1 happens at most four times.

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 $|V^{\varepsilon}_{st}|+1$ such r_i's by Lemma A.5(vi).

Case 3: $j = i+1$. As $r_i, r_{i+1} \in \mathrm{int}(e)$, the nodes r_i and r_{i+1} are not vertices of $\mathcal{T}$. Since the case that r_{i-1} is a vertex was already covered in case 1, we may assume that r_{i-1} is not a vertex of $\mathcal{T}$. By Lemma A.5(iv), (r_{i-1}, r_i, r_{i+1}) is a subpath of Q_0. We have $\overline{r_i r_{i+1}} \subset \mathrm{int}(e)$ as $r_i, r_{i+1} \in \mathrm{int}(e)$. Then, $\overline{r_{i-1} r_i}$ cannot be contained in e by Lemma A.4(i). So r_i is a critical node.

Case 4: There is no vertex of $\mathcal{T}$ in $R'([i,j])$, but there is a critical node r_k in $\mathrm{int}(R'[i,j])$. We charge r_i to r_k.

Case 5: $j > i+1$ and there is no vertex of $\mathcal{T}$ in $R'[i,j]$ and no critical node in $\mathrm{int}(R'[i,j])$. Lemma A.5(iv) implies that $R'[i,j]$ is a subpath of Q_0. By Lemma A.4(i), all of the nodes in $\mathrm{int}(R'[i,j])$ are transversal nodes. By Lemma A.6, we have $\mathrm{cost}(R'[i,j]) > \delta^{\varepsilon}_{st}/2$. Because $\mathrm{cost}(R') \leqslant \mathrm{cost}(P) + \varepsilon \mathsf{OPT}_{st}/2 \leqslant (2 + \varepsilon/2)\mathsf{OPT}_{st}$ (by Lemma A.5(ii) and our assumption on $\mathrm{cost}(P)$) and $\delta^{\varepsilon}_{st} = \varepsilon \mathsf{OPT}_{st}/(8\rho(|V^{\varepsilon}_{st}| + 2))$, we get

$$\mathrm{cost}(R'[i,j]) > \frac{\varepsilon}{8\rho(|V^{\varepsilon}_{st}| + 2)(4+\varepsilon)}\,\mathrm{cost}(R') \geqslant \frac{\varepsilon}{40\rho(|V^{\varepsilon}_{st}| + 2)}\,\mathrm{cost}(R').$$

So there are at most $40\rho(|V^{\varepsilon}_{st}| + 2)/\varepsilon$ such r_i's in N_e.

Cases 1, 2 and 5 contribute at most $40\rho(|V^{\varepsilon}_{st}| + 2)/\varepsilon + |V^{\varepsilon}_{st}| + 5$ nodes to N_e. The nodes in cases 3 and 4 are critical nodes outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$. So there are at most $4|V^{\varepsilon}_{st}| + 5$ of them by Lemma A.5(vi). Including the last node outside $\widehat{B}(s, \gamma^{\varepsilon}_{st})$ in $\mathrm{int}(e)$, we have $|N_e| \leqslant 40\rho(|V^{\varepsilon}_{st}| + 2)/\varepsilon + 5|V^{\varepsilon}_{st}| + 11$. The number of edges in $\mathcal{T}$ is at most $3(n+1) - 6 = 3n - 3$. Summing over all edges of $\mathcal{T}$ and including the vertices of $\mathcal{T}$ and the two endpoints of R', the number of nodes in R' is at most

$$n + 3 + (3n-3)(40\rho(|V^{\varepsilon}_{st}| + 2)/\varepsilon + 5|V^{\varepsilon}_{st}| + 11) < 3n(|V^{\varepsilon}_{st}| + 2)(40\rho/\varepsilon + 5) + 4n.$$

□

A.4 Proof of Lemma 2.2

Finally, we make use of Lemma A.5 and Lemma A.7 to prove Lemma 2.2. First, by a result in reference [6], there exists a polygonal path P that is a $(1+\varepsilon/2)$-approximate shortest path. The complexity of P could be extremely high though. We do the conversion by setting $R := \mathsf{Modify}(P, \varepsilon)$. By Lemma A.5(ii),

$$\mathrm{cost}(R) \leqslant \mathrm{cost}(P) + \varepsilon \mathsf{OPT}_{st}/2 \leqslant (1+\varepsilon)\mathsf{OPT}_{st}.$$

This shows that R is a $(1+\varepsilon)$-approximate shortest path from s to t.

To bound the number of links in R, let R_1 be the longest suffix of R whose nodes are outside $\widehat{B}(s, \gamma_{st}^{\varepsilon})$. By Lemma A.7, R_1 has at most $3n(|V_{st}^{\varepsilon}|+2)(40\rho/\varepsilon+5)+4n$ nodes. Let R_2 be the prefix of R that concatenates with R_1 to yield R. So the first node of R_2 is s and the last node of R_2 is in $\widehat{B}(s, \gamma_{st}^{\varepsilon})$.

Consider the output path Q_1 of the call $\mathsf{Bypass}(Q_0, \delta_{st}^{\varepsilon}, \gamma_{st}^{\varepsilon})$ in step 2 of Modify. Let Q_1' be the longest prefix of Q_1 whose nodes lie inside $\widehat{B}(s, \gamma_{st}^{\varepsilon})$. Then Q_1' is the geodesic path in $\widehat{B}(s, \gamma_{st}^{\varepsilon})$ from s to some vertex v of $\mathcal{T}$.

Let $r_k \in \widehat{B}(s, \gamma_{st}^{\varepsilon})$ be the node prior to R_1 on R. Since $\mathsf{Shortcut}$ in step 3 of Modify does not introduce any new nodes inside $\widehat{B}(s, \gamma_{st}^{\varepsilon})$, we have $r_k \in Q_1'$. For any other node r_l in R that is not a vertex of $\mathcal{T}$, $\mathsf{Shortcut}$ does not change the order of r_k and r_l along the path. Therefore, Q_1' contains all nodes on the subpath from s to r_k in R that are not vertices of $\mathcal{T}$. Recall that Q_1' has at most $4n-2$ nodes by Lemma A.3(i). There are at most $n+1$ nodes of R that are vertices of $\mathcal{T}$. Hence the number of nodes in the subpath from s to r_k in R is at most $5n-1$. In total, the number of nodes in R is fewer than $3n(|V_{st}^{\varepsilon}|+2)(40\rho/\varepsilon+5)+9n$.

Recall that we added t as a vertex to $\mathcal{T}$ by splitting the triangular face f containing t into three smaller triangles. The path R may not be a k_{st}^{ε}-link path in the original $\mathcal{T}$, where $k_{st}^{\varepsilon} = 3n(|V_{st}^{\varepsilon}|+2)\,(40\rho/\varepsilon+5)+9n$, because R may contain node(s) in the interior of f other than t. If r_i is such a node in $\mathrm{int}(f)$, we modify R by replacing the subpath (r_{i-1}, r_i, r_{i+1}) by (r_{i-1}, r_{i+1}). Clearly, the number of nodes decreases. Also, r_{i-1} and r_{i+1} lie in f as $r_i \in \mathrm{int}(f)$. So $\mathrm{cost}((r_{i-1}, r_i, r_{i+1})) \geqslant \mathrm{cost}(\overline{r_{i-1}r_{i+1}})$, which means that the path cost does not increase. We repeat the above until we obtain a k_{st}^{ε}-link path in the original $\mathcal{T}$.

B Additively weighted Voronoi diagram

This section contains the proofs of Lemma 2.3 and Lemma 2.4. It may be possible to improve the bound on the number of vertices in Lemma 2.3, but it does not seem straightforward. Dudley's construction [1, 10], for instance, gives an approximation with respect to the Hausdorff distance, but we need an approximation with respect to scaling. This type of approximation is more demanding: Dudley's construction gives a polygon with $O(1/\sqrt{\varepsilon})$ vertices, while the size of an approximation with respect to scaling cannot be bounded as a function of ε only. (See the appendix of Arya and Vigneron's research report on approximate Voronoi cells [4].) It may be possible to improve the bound in Lemma 2.3 from $O(\rho/\varepsilon)$ to $O(\rho/\sqrt{\varepsilon})$ using ideas from Dudley's construction. It would not improve the overall complexity bounds though, because we only compute Voronoi diagrams for cluster hubs, which are a small subset of our Steiner points, and so improving the Voronoi diagram computation does not affect the overall complexity bounds. The proof of Lemma 3.3 reveals that the space requirement and construction time of our data structures are dominated by the number of Steiner points placed in $\mathcal{T}$ and the running time of BUSHWHACK.

B.1 Proof of Lemma 2.3

Let f be a face of $\mathcal{T}$. Let o denote the origin at which the "unit disk" B_f of the convex distance function d_f is centered. By our assumption B_f is contained in a concentric unit Euclidean disk and B_f contains a concentric Euclidean disk with radius $1/\rho$.

We shoot rays from the origin o to $\mathrm{bd}(B_f)$ at angles $\varepsilon/(4\rho)$ apart. Given a point t such that $\|ot\| = 1$, the point t' on the ray from o to t that lies on $\mathrm{bd}(B_f)$ is given by $ot' = \frac{1}{d_f(o,t)} \cdot ot$. Thus, we can compute the intersections between the rays and $\mathrm{bd}(B_f)$ using the black-box model in which the distance between two given points under d_f can be returned in constant time.

Given a point t such that $\|ot\| = 1$, the point t' on the ray from o to t that lies on $\mathrm{bd}(B_f)$ is given by $ot' = \frac{1}{d_f(o,t)} \cdot ot$.

Let $\{x_1, x_2, \ldots\}$ denote the intersection points between these rays and $\mathrm{bd}(B_f)$ in counterclockwise order. We claim that the convex hull of $\{x_1, x_2, \ldots\}$ is the desired convex polygon C_f^ε. Clearly, C_f^ε lies inside B_f and C_f^ε has $O(\rho/\varepsilon)$ vertices. It remains to show that for any points $x, y \in \mathbb{R}^2$, we have $d_f(x,y) \leqslant d_f^\varepsilon(x,y) \leqslant (1+\varepsilon)d_f(x,y)$, where d_f^ε is the distance function induced by C_f^ε.

As $C_f^\varepsilon \subset B_f$, the inequality $d_f(x,y) \leqslant d_f^\varepsilon(x,y)$ follows directly from the definition of convex distance functions. To prove that $d_f^\varepsilon(x,y) \leqslant (1+\varepsilon)d_f(x,y)$, it suffices to show that $d_f^\varepsilon(o,y) \leqslant (1+\varepsilon)d_f(o,y)$ for any point $y \in \mathrm{bd}(B_f)$. Without loss of generality, assume that y lies in the cone bounded by rays from o through x_i and x_{i+1}. Let z be the intersection point $\overline{oy} \cap \overline{x_i x_{i+1}}$. Refer to Figure 11.

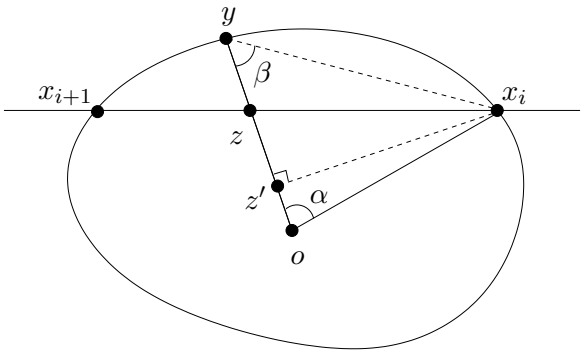

Figure 11: Illustration for the proof of $d_f^\varepsilon(o,y) \leqslant (1+\varepsilon)d_f(o,y)$ for $y \in \mathrm{bd}(B_f)$.

Let $\alpha = \angle x_i o y$ and $\beta = \angle x_i y o$. Since $\angle x_i o x_{i+1} \leqslant \frac{\varepsilon}{4\rho} \leqslant \frac{\varepsilon}{4}$, at least one of the perpendicular projections of x_i and x_{i+1}, say x_i, onto $\overline{oy}$ lies on the segment $\overline{oz}$. Denote this projection by z'. Then we have

$$\frac{\|yz\|}{\|oz\|} \leqslant \frac{\|yz'\|}{\|oz'\|} = \frac{\|ox_i\| \sin\alpha / \tan\beta}{\|ox_i\| \cos\alpha} = \frac{\tan\alpha}{\tan\beta}. \tag{3}$$

By the assumption on B_f and C_f^ε, the distance $\|oy\|$ is at most 1 and the distance from o to $\overline{x_i y}$ is at least $\frac{1}{\rho}\cos(\alpha/2)$. So we get $\sin\beta \geqslant \frac{\cos(\alpha/2)}{\rho}$, which implies that $\frac{1}{\tan\beta} \leqslant \frac{1}{\sin\beta} \leqslant \frac{\rho}{\cos(\alpha/2)}$. Since $\alpha \leqslant \frac{\varepsilon}{4\rho} < \pi/3$, we have $\cos(\alpha/2) \geqslant \cos\alpha > 1/2$ and thus that $\frac{1}{\tan\beta} \leqslant 2\rho$. Substituting the inequalities $\frac{1}{\tan\beta} \leqslant 2\rho$ and $\cos\alpha > 1/2$ into (3), we get

$$\frac{\|yz\|}{\|oz\|} \leqslant 2\rho\tan\alpha \leqslant 4\rho\sin\alpha \leqslant 4\rho\alpha \leqslant \varepsilon.$$

Hence

$$\frac{d_f^{\varepsilon}(o,y)}{d_f(o,y)} = \frac{\|oy\|}{\|oz\|} = 1 + \frac{\|yz\|}{\|oz\|} \leqslant 1 + \varepsilon.$$

B.2 Proof of Lemma 2.4

Voronoi diagrams under convex distance functions have been studied before [7, 15]. Ma [15] proved several geometric properties of such a Voronoi diagram in the unweighted case, including the connectedness of each Voronoi cell and the complexity of the bisectors. On the other hand, we are not aware of any published version of Lemma 2.4 in the literature.

We prove more general versions of Lemma 2.4(i) and (ii) for a convex distance function induced by a convex polygon. We first give a formal definition of the additively weighted Voronoi diagram. Notice that there is a tie-breaking rule based on the indices assigned to the sites.

Definition 10 *Let K be a positive integer. Let d be the convex distance function induced by a convex polygon C with K sides. Let S be a set of point sites $\{s_1, s_2, \ldots\}$ in $\mathbb{R}^2$. Each site $s_i \in S$ is associated with a weight $w_i \geqslant 0$ such that $w_i \leqslant w_j$ for any $i < j$.*

- *For any $i < j$, define $R_{ij} = \{x \in \mathbb{R}^2 : \mathrm{d}(s_i, x) + w_i \leqslant \mathrm{d}(s_j, x) + w_j\}$ and $R_{ji} = \mathbb{R}^2 \setminus R_{ij}$.*
- *For each site s_i, define the Voronoi cell of s_i to be $V_i = \cap_{j \neq i} R_{ij}$. Notice that $\mathrm{bd}(V_i)$ may not be a subset of V_i as V_i may not be a closed set.*
- *The additively weighted Voronoi diagram $\mathrm{Vor}(S, \mathrm{d})$ is the collection of the Voronoi cells and their boundary edges and vertices.*

The following result implies Lemma 2.4(i).

Lemma B.1 *For any Voronoi cell V_i in $\mathrm{Vor}(S, \mathrm{d})$, $s_i \in \mathrm{int}(V_i)$ and V_i is star-shaped.*

Proof. We first show that $s_i \in V_i$. If not, there must exist $j \neq i$ such that $w_j + \mathrm{d}(s_j, s_i) \leqslant w_i$, where the inequality is strict when $j > i$. Take a point $x \in V_i$. We have $w_i + \mathrm{d}(s_i, x) \geqslant w_j + \mathrm{d}(s_j, s_i) + \mathrm{d}(s_i, x) \geqslant w_j + \mathrm{d}(s_j, x)$, where the inequality is strict when $j > i$. But then x cannot belong to V_i by the tie-breaking rule, a contradiction.

Assume to the contrary that $s_i \in \mathrm{bd}(V_i)$. There must exist $j \neq i$ such that $w_i = w_j + \mathrm{d}(s_j, s_i)$, which is greater than w_j. If $i < j$, we get a contradiction because $w_i \leqslant w_j$ by definition. If $i > j$, then for any point $x \in V_i$, we have $w_i + \mathrm{d}(s_i, x) = w_j + \mathrm{d}(s_j, s_i) + \mathrm{d}(s_i, x) \geqslant w_j + \mathrm{d}(s_j, x)$, which implies that x cannot belong to V_i, a contradiction. This shows that $s_i \in \mathrm{int}(V_i)$.

The cell V_i is star-shaped if V_i contains $\overline{s_i x}$ for any point $x \in V_i$. Assume to the contrary that there exists a point $y \in \overline{s_i x}$ such that $y \in V_j$ for some $j \neq i$. It follows from the definition of $x \in V_i$ that $w_i + \mathrm{d}(s_i, x) \leqslant w_j + \mathrm{d}(s_j, x)$, where the inequality is strict when $j < i$. We have $w_i + \mathrm{d}(s_i, y) = w_i + \mathrm{d}(s_i, x) - \mathrm{d}(y, x) \leqslant w_j + \mathrm{d}(s_j, x) - \mathrm{d}(y, x)$, which is at most $w_j + \mathrm{d}(s_j, y)$ by the triangle inequality. So $w_i + \mathrm{d}(s_i, y) \leqslant w_j + \mathrm{d}(s_j, y)$ and the inequality is strict when $j < i$. But then y cannot belong to V_j by the tie-breaking rule, a contradiction. □

It follows from Lemma B.1 that there are at most $|S|$ Voronoi cells in $\mathrm{Vor}(S, \mathrm{d})$ and each Voronoi cell is connected. So the dual graph of $\mathrm{Vor}(S, \mathrm{d})$ is a planar graph with at most $|S|$ graph vertices. By Euler's relation, there are $O(|S|)$ faces in any planar embedding of this graph, which implies that there are $O(|S|)$ Voronoi vertices in $\mathrm{Vor}(S, \mathrm{d})$. It remains to analyze the complexity of the Voronoi edges in $\mathrm{Vor}(S, \mathrm{d})$ to fully determine the complexity of $\mathrm{Vor}(S, \mathrm{d})$.

Lemma B.2 *Let s_i and s_j be two points in S such that $i < j$. The additively weighted bisector $\mathrm{bd}(R_{ij})$ of s_i and s_j under the distance function d is a polygonal chain with at most $2K$ vertices.*

Proof. We go to $\mathbb{R}^3$ and apply a lifting argument. Let the plane containing S be the horizontal plane H through the origin. Let E_i be the vertical convex cone with apex s_i above the plane of S such that for any point $x \in H$, the vertical distance from x to E_i is equal to $\mathrm{d}(s_i, x)$. This implies that any horizontal cross-section of E_i is a homothetic copy of the convex polygon C. So E_i has exactly K unbounded triangular faces.

We translate E_i vertically upward by a distance w_i. Denote the translated cone by F_i. Now, for any point $x \in H$, the vertical distance from x to F_i is equal to $w_i + \mathrm{d}(s_i, x)$. We similarly define F_j for s_j. Any point x in $\mathrm{bd}(R_{ij})$ has the same vertical distance from F_i and F_j. It follows that x lifts vertically to an intersection point z between a face σ_i of F_i and a face τ_j of F_j.

Conversely, take a face σ_i of F_i. If σ_i avoids F_j, then the lifting of any point $x \in H$ below σ_i hits σ_i before F_j. So $x \in \mathrm{int}(R_{ij})$. In other words, σ_i does not contribute any point to $\mathrm{bd}(R_{ij})$. If σ_i intersects a face τ_j and σ_i is coplanar with τ_j, then σ_i does not contribute any point to $\mathrm{bd}(R_{ij})$ due to the tie-breaking rule. The remaining case is that σ_i intersects F_j and σ_i is not coplanar with any face of F_j that it intersects. In this case, $\sigma_i \cap F_j$ is a polygonal chain in $\mathbb{R}^3$ and the projection of this chain is part of $\mathrm{bd}(R_{ij})$. So a vertex of $\sigma_i \cap F_j$ corresponds to a vertex of $\mathrm{bd}(R_{ij})$. Therefore, to bound the complexity of $\mathrm{bd}(R_{ij})$, it suffices to bound the total complexity of $\sigma_i \cap F_j$ over all faces σ_i of F_i.

Each vertex of $\sigma_i \cap F_j$ is the isolated intersection point between an edge e_j of F_j and σ_i. Since F_i is convex, the edge e_j can produce at most two isolated intersection points with the faces of F_i. This implies that $\mathrm{bd}(R_{ij})$ has at most $2K$ vertices. ◻

Lemma B.2 allows us to conclude that $\mathrm{Vor}(S, \mathrm{d})$ has linear complexity which proves the Voronoi diagram complexity in Lemma 2.4(ii).

Corollary 3 *The complexity of $\mathrm{Vor}(S, \mathrm{d})$ is $O(K|S|)$.*

It remains to show how to construct $\mathrm{Vor}(S, \mathrm{d})$ efficiently. Chew and Drysdale [7] proposed a divide-and-conquer algorithm for constructing additively weighted Voronoi diagrams. The running time of this algorithm is $O(K|S| \log |S|)$, which is exactly what we needed. However, the conquer step of this divide-and-conquer algorithm requires an assumption, which we explain below.

Divide S into two equal subsets S_L and S_R using the median x-coordinate. Let $B(S_L, S_R, \mathrm{d})$ denote the set of Voronoi edges of $\mathrm{Vor}(S, \mathrm{d})$ such that, among the two cells incident on each edge in $B(S_L, S_R, \mathrm{d})$, one is owned by some point in S_L and the other is owned by some point in S_R. The divide step is to recursively compute $\mathrm{Vor}(S_L, \mathrm{d})$ and $\mathrm{Vor}(S_R, \mathrm{d})$ and then merge them by tracing $B(S_L, S_R, \mathrm{d})$. For the tracing to work properly, it is essential that $B(S_L, S_R, \mathrm{d})$ does not contain any loop. This is the assumption needed for the algorithm of Chew and Drysdale to work correctly.

If the point sites in S are in arbitrary positions, it is possible that $B(S_L, S_R, \mathrm{d})$ contains loops. However, we show below that if the sites in S are collinear, $B(S_L, S_R, \mathrm{d})$ is loop-free.

Lemma B.3 *Assume that the sites in S are collinear. Let (S_L, S_R) be a partition of S such that the points in S_L and S_R are contiguous. Then, $B(S_L, S_R, \mathrm{d})$ does not contain any loop.*

Proof. Assume that S lies on a horizontal line ℓ and that S_L is to the left of S_R along ℓ. Assume to the contrary that there is a loop in $B(S_L, S_R, \mathrm{d})$. Without loss of generality, we can

pick a loop in $B(S_L, S_R, \mathrm{d})$ that is innermost in the sense that this loop does not enclose any portion of $B(S_L, S_R, \mathrm{d})$.

This loop encloses a portion of the Voronoi diagram of S_L or S_R, and the Voronoi diagram of the other subset is outside the loop. Without loss of generality, assume that the loop encloses a portion of $\mathrm{Vor}(S_L, \mathrm{d})$ and $\mathrm{Vor}(S_R, \mathrm{d})$ lies outside the loop.

Pick a Voronoi cell V_l of $\mathrm{Vor}(S, \mathrm{d})$ inside the loop. So s_l belongs to S_L. Pick an intersection point x between the loop and the ray shooting from s_l to the left. Let s_r be the site in S_R whose additively weighted distance to x is minimum. Since x does not belong to $\mathrm{int}(V_l)$, we get

$$w_r + \mathrm{d}(s_r, x) \leqslant w_l + \mathrm{d}(s_l, x).$$

Observe that x lies to the left of s_l by our choice, s_l lies to the left of s_r by assumption, and the three points are collinear. Thus, $\mathrm{d}(s_r, x) = \mathrm{d}(s_r, s_l) + \mathrm{d}(s_l, x)$. Substituting this into the inequality above, we get $w_r + \mathrm{d}(s_r, s_l) + \mathrm{d}(s_l, x) \leqslant w_l + \mathrm{d}(s_l, x)$, which implies that $w_r + \mathrm{d}(s_r, s_l) \leqslant w_l$. Thus, s_l cannot lie in the interior of R_{lr}. But then s_l cannot lie in the interior of $V_l \subseteq R_{lr}$, contradicting Lemma B.1. □

Lemma B.3 allows us to apply the algorithm of Chew and Drysdale to construct $\mathrm{Vor}(S, \mathrm{d})$ in $O(K|S| \log |S|)$ time when the sites in S are collinear. This completes the proof of Lemma 2.4(ii).

The proof of Lemma 2.4(iii) is simple. By Lemma 2.3(ii), for any s_j, we have

$$\begin{aligned} w_j + d_f(s_j, x) &\geqslant w_j + \frac{d_f^\varepsilon(s_j, x)}{1+\varepsilon} \\ &\geqslant \frac{w_j + d_f^\varepsilon(s_j, x)}{1+\varepsilon} \\ &\geqslant \frac{\min_{s_i \in S}\{w_i + d_f^\varepsilon(s_i, x)\}}{1+\varepsilon}. \end{aligned}$$

Since the above inequality holds for any s_j, we conclude that $\min_{s_i \in S}\{w_i + d_f^\varepsilon(s_i, x)\} \leqslant (1+\varepsilon)\min_{s_i \in S}\{w_i + d_f(s_i, x)\}$. This proves Lemma 2.4(iii).