Discrete & Computational Geometry (2023) 70:426–454
https://doi.org/10.1007/s00454-022-00424-6

An Optimal Deterministic Algorithm for Geodesic Farthest-Point Voronoi Diagrams in Simple Polygons

Haitao Wang[1]

Received: 29 April 2021 / Revised: 21 February 2022 / Accepted: 6 March 2022 /
Published online: 30 August 2022

Abstract
Given in the plane a set S of m point sites in a simple polygon P of n vertices, we consider the problem of computing the geodesic farthest-point Voronoi diagram for S in P. It is known that the problem has an $\Omega(n+m\log m)$ time lower bound. Previously, a randomized algorithm was proposed [Barba, SoCG 2019] that solves the problem in $O(n+m\log m)$ expected time. The previous best deterministic algorithms solve the problem in $O(n\log\log n+m\log m)$ time [Oh, Barba, and Ahn, SoCG 2016] or in $O(n+m\log m+m\log^2 n)$ time [Oh and Ahn, SoCG 2017]. In this paper, we present a deterministic algorithm that takes $O(n+m\log m)$ time, which is optimal. This answers affirmatively an open question posed by Mitchell in the Handbook of Computational Geometry two decades ago.

Keywords Farthest sites · Voronoi diagrams · Geodesic distance · Shortest paths · Simple polygons

Mathematics Subject Classification 68Q25 · 68W40 · 68U05

1 Introduction

Let P be a simple polygon of n vertices in the plane. Let S be a set of m points, called *sites*, in P (each site can be either in the interior or on the boundary of P). For any two points in P, their *geodesic distance* is the length of their Euclidean shortest path in P.

Editor in Charge: Kenneth Clarkson

This research was supported in part by NSF under Grant CCF-2005323. A preliminary version of this paper appeared in Proceedings of the 37th International Symposium on Computational Geometry (SoCG 2021).

Haitao Wang
haitao.wang@usu.edu

[1] Department of Computer Science, Utah State University, Logan, UT 84322, USA

We consider the problem of computing the *geodesic farthest-point Voronoi diagram* of S in P, which is to partition P into Voronoi cells such that all points in the same cell have the same farthest site in S with respect to the geodesic distance.

This problem generalizes the Euclidean farthest Voronoi diagram of m sites in the plane, which can be computed in $O(m \log m)$ time [26]; this is optimal as $\Omega(m \log m)$ is a lower bound [18], e.g., by a reduction from determining the convex hull of a set of points in the plane [5, 12, 30]. For the more general geodesic problem in P, Aronov et al. [3] showed that the complexity of the diagram is $\Theta(n + m)$ and provided an $O(n \log n + m \log m)$ time algorithm. The runtime is close to optimal as $\Omega(n + m \log m)$ is a lower bound. No progress had been made for over two decades until in SoCG 2016 Oh et al. [23] proposed an $O(n \log \log n + m \log m)$ time algorithm. Later in SoCG 2017 Oh and Ahn [22] gave another $O(n + m \log m + m \log^2 n)$ time algorithm and in SoCG 2019 Barba [9] presented a randomized algorithm that solves the problem in $O(n + m \log m)$ expected time.

In this paper, we give an $O(n + m \log m)$ time deterministic algorithm, which is optimal. The space complexity of the algorithm is $O(n + m)$, which is optimal. This answers affirmatively an open question posed by Mitchell [20] in the Handbook of Computational Geometry two decades ago, whether an $O(n + m \log m)$ time deterministic algorithm exists for constructing the geodesic farthest-point Voronoi diagram for S in P.

1.1 Related Work

If all sites of S are on the boundary of P, then better results exist. The algorithm of Oh et al. [23] solves the problem in $O((n + m) \log \log n)$ time while the randomized algorithm of Barba [9] runs in $O(n + m)$ expected time.

The *geodesic nearest-point Voronoi diagram* for point sites in a simple polygon has also attracted much attention. The problem also has an $\Omega(n + m \log m)$ time lower bound. The first close-to-optimal algorithm was given by Aronov [2] with a running time of $O((n + m) \log(n + m) \log n)$. Papadopoulou and Lee [24] improved the algorithm's runtime to $O((n + m) \log(n + m))$. Recent progress has been made by Oh and Ahn [22] who presented an $O(n + m \log m \log^2 n)$ time algorithm and also by Liu [19] who designed an $O(n + m(\log m + \log^2 n))$ time algorithm. Finally the problem was solved optimally in $O(n + m \log m)$ time by Oh [21].

Another closely related problem is to compute the *geodesic center* of a simple polygon P, which is a point in P that minimizes the maximum geodesic distance from all points of P. Asano and Toussaint [4] first gave an $O(n^4 \log n)$ time algorithm for the problem. Pollack et al. [25] derived an $O(n \log n)$ time algorithm. Recently the problem was solved optimally in $O(n)$ time by Ahn et al. [1]. The *geodesic diameter* of P is the largest geodesic distance between any two points in P. Chazelle [10] first gave an $O(n^2)$ time algorithm and then Suri [27] presented an improved $O(n \log n)$ time solution. Hershberger and Suri [15] finally solved the problem in $O(n)$ time, which is optimal.

All above results are for simple polygons. For polygons with holes, the problems become more difficult. The geodesic nearest-point Voronoi diagram for m point sites in a polygon with holes of n vertices can be solved in $O((n+m)\log(n+m))$ time by the algorithm of Hershberber and Suri [16]. Bae and Chwa [6] gave an algorithm for constructing the geodesic farthest-point Voronoi diagram and the algorithm runs in $O(nm\log^2(n+m)\log m)$ time. For computing the geodesic diameter in a polygon with holes, Bae et al. [7] solved the problem in $O(n^{7.73})$ or $O(n^7(h+\log n))$ time, where h is the number of holes. For computing the geodesic center, Bae et al. [8] first gave an $O(n^{12+\epsilon})$ time algorithm, for any constant $\epsilon > 0$; Wang [29] presented an improved algorithm of $O(n^{11}\log n)$ time.

1.2 Our Approach

We follow the algorithmic scheme in [22], which in turn follows that in [3]. Specifically, we first compute the geodesic convex hull of all sites of S in $O(n+m\log m)$ time [13, 14, 28], and then compute the geodesic center c^* of the hull in $O(n+m)$ time [1]. Aronov [3] showed that the farthest-point Voronoi diagram forms a tree embedded inside P with c^* as the root and all leaves on ∂P, the boundary of P. We construct the farthest-point Voronoi diagram restricted to ∂P; this can be done in $O(n+m)$ time by a recent algorithm of Oh et al. [23] once the geodesic convex hull of S is known.

Next we run a *reverse geodesic sweeping* algorithm to extend the diagram from ∂P to the interior of P (i.e., based on all leaves on ∂P and the root c^* of the tree, we want to construct the tree). Here we use a *geodesic sweeping circle* that consists of all points with the same geodesic distance from c^*. Aronov [3] implemented this sweeping algorithm in $O((n+m)\log(n+m))$ time. Oh and Ahn [22] gave an improved solution of $O(n+m\log m+m\log^2 n)$ time by using a data structure for the following query problem: Given three points in P, compute the point that is equidistant from them. Oh and Ahn [22] built a data structure in $O(n)$ time that can answer each query in $O(\log^2 n)$ time, and that is why the time complexity of their algorithm has a $\log^2 n$ factor. We improve the query time to $O(\log n)$ (with $O(n)$ time preprocessing) with the help of the following observations. First, the three points involved in a query are three sites of S whose Voronoi cells are adjacent along the sweeping circle. Second, among the three sites involved in a query, for every two sites whose Voronoi cells are adjacent, the sweeping algorithm provides us with a point equidistant to them. These observations along with the tentative prune-and-search technique of Kirkpatrick and Snoeyink [17] lead us to a query algorithm of $O(\log n)$ time. Consequently, the sweeping algorithm can be implemented in $O(n+m\log m)$ time.

We should point out that in her algorithm for computing the geodesic nearest-point Voronoi diagram, Oh [21] also announced an $O(\log n)$ time algorithm for the above query problem and her algorithm also uses the tentative prune-and-search technique (although the details are omitted due to the page limit). However, the difference is that she uses a balanced geodesic triangulation [11] and her result is based on the assumption that the sought point of the query lies in a known geodesic triangle $\triangle$ and the three query points are in the same subpolygon of P separated by a side of $\triangle$ (see [21, Lemma 4.2]). For our problem, we do not need the balanced geodesic triangulation and do not have such an assumption. Instead, our algorithm relies on the observations mentioned above.

The rest of the paper is organized as follows. Section 2 defines notation and introduces some concepts. The algorithm for constructing the geodesic Voronoi diagram is described in Sect. 3. Section 4 presents the algorithm for a lemma about the query problem discussed above.

2 Preliminaries

Like the previous work [3, 9, 22, 23], for ease of discussion, we make a general position assumption that no vertex of P is equidistant from two sites of S and no point of P has four farthest sites. We occasionally use *polygon vertex* to refer to a vertex of P and use *polygon edge* to refer to an edge of P.

For any two points p and q in P, let $\pi(p,q)$ denote the (Euclidean) shortest path from p to q in P; let $d(p,q)$ denote the length of $\pi(p,q)$. $\pi(p,q)$ is also called the *geodesic path* and $d(p,q)$ is called the *geodesic distance* between p and q. The vertex of $\pi(p,q)$ adjacent to q (resp., p) is called the *anchor* of q (resp., p) in $\pi(p,q)$.

For any two points a and b in the plane, denote by $\overline{ab}$ the line segment with a and b as endpoints, and denote by $|\overline{ab}|$ the length of the segment.

For any two sites s and t of S, their *bisector*, denoted by $B(s,t)$, consists of all points of P equidistant from them, i.e., $B(s,t) = \{p \mid d(s,p) = d(t,p),\ p \in P\}$. Due to the general position assumption, Aronov et al. [2] showed that $B(s,t)$ is a smooth curve connecting two points on ∂P with no other points common with ∂P and $B(s,t)$ comprises $O(n)$ straight and hyperbolic arcs (a straight arc is a line segment); the endpoints of the arcs are *breakpoints*, each of which is the intersection of $B(s,t)$ and a half-line extended from a polygon vertex u to another polygon vertex v such that u is an anchor of v in $\pi(s,v)$ or in $\pi(t,v)$ (it is possible that $u=s$ or $u=t$); e.g., see Fig. 1.

For any site $s \in S$, define $C(s)$ as the region consisting of all points p of P whose farthest site is s, i.e., $C(s) = \{p \mid d(p,s) \ge d(p,s'),\ s' \in S\}$; e.g., see Fig. 2. We call $C(s)$ the (farthest) *Voronoi cell* of s. Note that $C(s)$ may be empty; if $C(s)$ is not empty, then it is simply connected [3]. The Voronoi cells of all sites of S form a partition of P. We define the *geodesic farthest-point Voronoi diagram* (or *farthest Voronoi diagram* for short), denoted by $\mathrm{FVD}(S)$, as the closure of the interior of P minus the union of the interior of $C(s)$ for all $s \in S$; alternatively, $\mathrm{FVD}(S) = \{p \in B(s,t) \mid s,t \in S \text{ and } d(s,p) = \max_{r \in S} d(r,p)\}$; e.g., see Fig. 2. A point v of $\mathrm{FVD}(S)$ is a *Voronoi vertex* if it is an intersection of a bisector with ∂P or if it has degree 3 (i.e., it has three equidistant sites). The curve of $\mathrm{FVD}(S)$ connecting two adjacent vertices is called a *Voronoi edge*, which is a portion of a bisector of two sites. Note that a Voronoi edge may not be of constant size because it may contain multiple breakpoints. While $\mathrm{FVD}(S)$ has $O(m)$ Voronoi vertices and edges, the total complexity of $\mathrm{FVD}(S)$ is $O(n+m)$ [3].

A subset P' of P is *geodesically convex* if $\pi(p,q)$ is in P' for any two points p and q in P'. The *geodesic convex hull* of S in P, denoted by $\mathrm{GCH}(S)$, is the common intersection of all geodesically convex sets containing S (e.g., see Fig. 3). $\mathrm{GCH}(S)$ is a weakly simple polygon of at most $n+m$ vertices. Let c^* be the geodesic center of $\mathrm{GCH}(S)$, which is also the geodesic center of S [3]. Note that c^* must be on a Voronoi

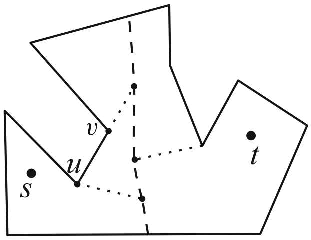

Fig. 1 Illustrating the bisector $B(s, t)$ (the dashed curve) with three breakpoints. The extension from u to v defines a break point

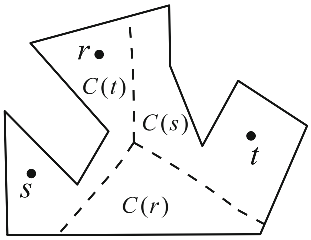

Fig. 2 Illustrating a geodesic farthest-point Voronoi diagram for three points s, t, and r

edge of FVD(S). Indeed, if c^* has three farthest sites in S, then c^* is a Voronoi vertex; otherwise it has two farthest sites and thus is in the interior of an edge of FVD(S). Aronov [3] proved that FVD(S) is a tree with c^* as the root and all leaves on ∂P; he also showed that only sites on the boundary of GCH(s) have nonempty cells in FVD(S) and the ordering of the sites with nonempty cells around the boundary of GCH(s) is the same as the ordering of their Voronoi cells around ∂P (the *ordering lemma*). Note that a site on the boundary of GCH(s) may still have an empty Voronoi cell and intuitively this is because P is not large enough [3].

Consider any three points s, t, r in P. The vertex farthest to s in $\pi(s, t) \cap \pi(s, r)$ is called the *junction vertex* of $\pi(s, t)$ and $\pi(s, r)$. The closure of the interior of the geodesic convex hull GCH(s, t, r) is called the *geodesic triangle* of s, t, and r, denoted by $\triangle(s, t, r)$, whose boundary is composed of three convex chains $\pi(s', t')$, $\pi(t', r')$, $\pi(r', s')$, where s' is the junction vertex of $\pi(s, t)$ and $\pi(s, r)$, and t' and r' are defined likewise; e.g., see Fig. 4. The three convex chains are called *sides* of $\triangle(s, t, r)$. The three vertices s', t', and r' are called the *apexes* of $\triangle(s, t, r)$.

3 Computing the Farthest Voronoi Diagram FVD(S)

In this section, we present our algorithm for computing the farthest Voronoi diagram FVD(S). First, we compute the geodesic convex hull GCH(S) of S in $O(n+m \log m)$ time [13, 14, 28]. Second, we compute the geodesic center c^* of GCH(S) in $O(n+m)$ time [1]. Third, we compute the portion of FVD(S) restricted to the polygon boundary ∂P, i.e., the leaves of FVD(S). This can be done in $O(n+m)$ time by the algorithm

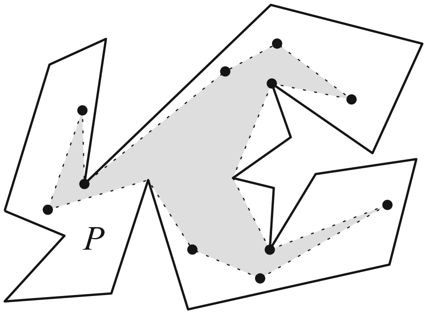

Fig. 3 Illustrating the geodesic convex hull (the grey region) for a set of points in P

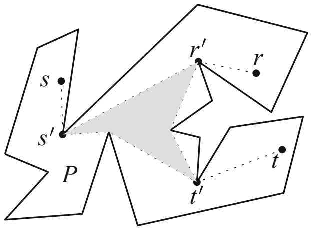

Fig. 4 Illustrating a geodesic triangle $\triangle(s, t, r)$ (the grey region)

in [23].[1] The fourth step is to extend the diagram to the interior of P, i.e., construct the tree FVD(S) based on all its leaves and the root c^*. This is achieved by a reverse geodesic sweeping algorithm, whose details are described below.

The algorithm first computes the adjacency information of FVD(S). Specifically, we will compute the locations of all Voronoi vertices of FVD(S); for Voronoi edges, however, we will not compute them exactly (i.e., the locations of their breakpoints will not be computed) but only output their incident Voronoi vertices, i.e., if u and v are the two Voronoi vertices incident to the same Voronoi edge, then we will output the pair (u, v) as an *abstract* Voronoi edge. In this way, we will output the abstract tree FVD(S) with the exact locations of all Voronoi vertices; this is called the *topological structure* of FVD(S) in [22]. After having the topological structure, Oh and Ahn [22] gave an algorithm that can construct FVD(S) in additional $O(n+m\log m)$ time. More specifically, with $O(n)$ time preprocessing, each Voronoi edge can be computed in $O(\log n+k)$ time, where k is the number of breakpoints in the Voronoi edge (see [22, Sect. 4] for details). As FVD(S) has $O(m)$ Voronoi edges, the total time for computing all Voronoi edges is $O(m\log n + K)$, where K is the total number of breakpoints on all Voronoi edges. As $K = O(n+m)$ [3], the total time is bounded by $O(n+m\log n)$,

[1] Note that the result was not explicitly given in [23] but can be obtained from their $O(n\log\log n+m\log m)$ time algorithm for computing FVD(S). Indeed, given GCH(S), the algorithm first partitions P in $O(n+m)$ time into $O(1)$ subpolygons such that each subpolygon P' is for a problem instance where all involved sites are on the boundary of P' (see [23, Sect. 7]). Then, each problem instance is further reduced in linear time to a problem instance where all sites are vertices of P' (see [23, Sect. 6]), and each such problem instance can be solved in linear time (see [23, Sect. 3]). The total running time of all above is $O(n + m)$ (for computing FVD(S) restricted to the boundary of P only). This result was also used by Oh and Ahn [22] in their $O(n + m\log m + m\log^2 n)$ time algorithm for computing FVD(S).

which is $O(n+m \log m)$.[2] In the following, we will focus on computing the topological structure of FVD(S).

We use a reverse geodesic sweeping as in [3, 22]. Roughly speaking, the sweep line is a *geodesic circle* C consisting of all points in P that have the same geodesic distance from the geodesic center c^* of S. This statement is actually not quite accurate as initially the sweep circle is just the boundary of P. During the sweep, we maintain the sites whose Voronoi cells currently intersect C; these sites are stored in a cyclic linked list $\mathcal{L}$ ordered by their intersections with C. Initially when $C = \partial P$, as we already have the leaves of FVD(S), we can build $\mathcal{L}$ in $O(n+m)$ time. Note that $|\mathcal{L}| = O(m)$. During the algorithm, C will shrink until arriving at c^*; an event happens when C hits a Voronoi vertex, which will be computed on the fly. Specifically, for each triple of adjacent sites s, t, r in the list $\mathcal{L}$, we compute the point, denoted by $\alpha(s, t, r)$, equidistant from them, which is the intersection of the bisectors $B(s, t)$ and $B(t, r)$. Due to our general position assumption, $\alpha(s, t, r)$ is unique if it exists (see [3, Lemma 2.5.3]). We store all these α-points in a priority queue Q, ordered by decreasing geodesic distance from c^*. In order to compute the α-points, for any pair of adjacent sites s and t in $\mathcal{L}$, we maintain a Voronoi vertex, denoted by $\beta(s, t)$, on their bisector $B(s, t)$ with the following property: $\beta(s, t)$ is outside or on the current geodesic circle C. Initially, we set $\beta(s, t)$ to be the Voronoi vertex on ∂P incident to the Voronoi cells of s and t; so the above property holds as $C = \partial P$.

The main loop of the algorithm works as follows. As long as Q is not empty, we repeatedly extract the point with largest geodesic distance from c^* and let the point be $\alpha(s, t, r)$ defined by three sites s, t, r in this order in $\mathcal{L}$. We report $\alpha(s, t, r)$ as a Voronoi vertex and report $(\beta(s, t), \alpha(s, t, r))$ and $(\beta(t, r), \alpha(s, t, r))$ as two abstract Voronoi edges. We remove t from $\mathcal{L}$ and set $\beta(s, r) = \alpha(s, t, r)$. Let x be the neighbor of s other than r in $\mathcal{L}$ and let y be the neighbor of r other than s. We remove $\alpha(x, s, t)$ and $\alpha(t, r, y)$ from Q if they exist. Next, we compute $\alpha(x, s, r)$ and $\alpha(s, r, y)$ (if they exist) as well as their geodesic distances from c^*, and insert them into Q.

For the running time, there are $O(m)$ events, because the total number of Voronoi vertices of FVD(S) is $O(m)$ [3], and thus the total time of the algorithm is $O(m(\sigma + \log m))$, where $O(\sigma)$ is the time for computing each α-point. Lemma 3.1, which will be proved later in Sect. 4, is for computing the α-points.

Lemma 3.1 *With $O(n)$ time preprocessing, for any triple of adjacent sites s, t, r in $\mathcal{L}$ at any moment during the algorithm, given the two Voronoi vertices $\beta(s, t)$ and $\beta(t, r)$, our algorithm can do the following in $O(\log n)$ time: if $\alpha(s, t, r)$ is a Voronoi vertex, then compute it; otherwise, either compute $\alpha(s, t, r)$ or return null.*

We remark that Lemma 3.1 is sufficient for the correctness of our geodesic sweeping algorithm as only Voronoi vertices are essential. If the algorithm returns null, the event will not be inserted to Q. With Lemma 3.1 at hand, our geodesic sweeping algorithm computes the topological structure of FVD(S) in $O(n + m \log m)$ time. After that, as discussed above, we can compute the full diagram FVD(S) in additional $O(n + m \log m)$ time by the techniques of Oh and Ahn [22]. Also, the space of the algorithm is bounded by $O(n + m)$.

[2] Indeed, if $m < n/\log n$, then $n+m \log n = \Theta(n)$, which is $O(n+m \log m)$; otherwise, $\log n = O(\log m)$ and $n + m \log n = O(n + m \log m)$.

Theorem 3.2 *The geodesic farthest-point Voronoi diagram of a set of m points in a simple polygon of n vertices can be computed in* $O(n + m \log m)$ *time and* $O(n + m)$ *space.*

4 Algorithm for Lemma 3.1

In this section, we present our algorithm for Lemma 3.1. We first present an algorithm in Sect. 4.1 for the following *triple-point geodesic center query* problem: given any three points in P, compute their geodesic center in P, which is a point that minimizes the largest geodesic distance from the three query points. Oh and Ahn [22] solved this problem in $O(\log^2 n)$ time, after $O(n)$ time preprocessing. Our algorithm runs in $O(\log n)$ time also with $O(n)$ time preprocessing.[3] This algorithm will be used as a subroutine in our algorithm for Lemma 3.1, which will be discussed in Sect. 4.2.

4.1 The Triple-Point Geodesic Center Query Problem

For preprocessing, we construct the two-point shortest path query data structure by Guibas and Hershberger [13, 14] and we refer to it as the *GH data structure*. The data structure can be constructed in $O(n)$ time, after which given any two points p and q in P, the geodesic distance $d(p, q)$ can be computed in $O(\log n)$ time and the geodesic path $\pi(p, q)$ can be output in additional time linear in the number of edges of $\pi(p, q)$.

Consider three query points s, t, and r in P. Our goal is to compute their geodesic center, denoted by c. We follow the algorithmic scheme in [22]. Consider the geodesic convex hull $\mathrm{GCH}(s, t, r)$ and the geodesic triangle $\triangle(s, t, r)$. We know that c is the geodesic center of $\mathrm{GCH}(s, t, r)$ [3]. Depending on whether c is in the interior of $\triangle(s, t, r)$, there are two cases.

4.1.1 c is not in the Interior of $\triangle(s, t, r)$

If c is not in the interior of $\triangle(s, t, r)$, then it must be on the geodesic path of two points of $\{s, t, r\}$. Without loss of generality, we assume that $c \in \pi(s, t)$. Note that c must be the middle point of $\pi(s, t)$. To locate c in $\pi(s, t)$, we wish to do binary search on the vertices of $\pi(s, t)$. It was claimed in [22] that the query algorithm of the GH data structure returns $\pi(s, t)$ as a binary tree (so that binary search can be done in a straightforward way), in particular, when the simpler approach in [14] is utilized. In fact, this is not quite precise. Indeed, the binary tree structures in both [13] and [14] are used for representing convex chains (or more rigorously, *semiconvex chains* [14]). However, $\pi(s, t)$ is actually a *string* [13], which in general is not a semiconvex chain. The data structure for representing a string is a tree but not necessarily a binary tree because a node in the tree may have three children.

[3] To be fair, this problem is not a dominant one in their algorithm, which might be a reason Oh and Ahn [22] did not push their result further.

Here for completeness, we provide a general binary search scheme on the geodesic path $\pi(p, q)$ returned by the GH data structure for any two query points p and q in P. Suppose we are looking for either a vertex or an edge of $\pi(p, q)$, denoted by w^* in either case, and we have access to an *oracle* such that given any vertex $v \in \pi(p, q)$, the oracle can determine whether w^* is in $\pi(p, v)$ or in $\pi(v, q)$. Then, we have the following lemma.

Lemma 4.1 *With $O(n)$ time preprocessing, given any two query points p and q, the sought vertex or edge w^* can be located by a binary search algorithm that calls the oracle on $O(\log n)$ vertices of $\pi(p, q)$, and the total time of the binary search excluding the time for calling the oracle is $O(\log n)$. In particular, the middle point of $\pi(p, q)$ can be found in $O(\log n)$ time.*

Proof We will use notation and concepts from the GH data structure [13] without much explanation. To represent convex chains, we utilize the simpler way given in [14], i.e., persistent binary trees with the path-copying method.

Consider the query points p and q. The GH query algorithm combines $O(\log \log n)$ "small" *hourglasses* of size $O(\log n)$ and two "big" hourglasses of size $O(n)$ to assemble the path $\pi(p, q)$, which is represented as a *string* [13]. Combining two hourglasses involves computing a tangent between them such that the tangent belongs to $\pi(p, q)$. Thus, the algorithm will produce $O(\log \log n)$ tangents. For our problem, we explicitly consider these tangents and call the oracle on every vertex of these tangents. This calls the oracle $O(\log \log n)$ times. After that we can determine an hourglass containing w^*. If it is a small hourglass, then since it has $O(\log n)$ vertices, we can simply call the oracle on every vertex to locate w^*. In the following, we assume that w^* is in a big hourglass and let π be the portion of $\pi(p, q)$ in the hourglass.

The endpoints of π can be obtained during the GH query algorithm. π from one end to the other consists of a convex chain, a string, and another convex chain in order. The two connection vertices between the string and the two convex chains can be maintained during the preprocessing. We call the oracle on these two vertices, after which we can determine the one of the three portions of π that contains w^*. If it is a convex chain, then as a convex chain is represented by a binary tree of height $O(\log n)$ [14], we can apply binary search on this tree in a standard way; after calling the oracle on $O(\log n)$ vertices, w^* can be obtained. In the following, we assume that the string of π contains w^*. By slightly abusing notation, we still use π to denote the string.

The string π is represented by a tree T. However, each node of T may have three children. Consider the root v of T. In general, π is a *derived string* and v has three children: a left subtree L representing a derived string, a middle subtree M representing a *fundamental string*, and a right subtree R representing another derived string. The fundamental string consists of two convex chains linked by a tangent edge and each derived string consists of two derived strings and a fundamental string in the middle. The height of T is $O(\log n)$. The connecting vertex between the left string and the middle string and the connecting vertex between the middle string and the right string are maintained in the preprocessing and thus available during the query algorithm. We call the oracle on the two connecting vertices and then determine which string contains w^*. If the left or the right string contains w^*, then we proceed on the corresponding subtree of v recursively. Otherwise, the middle string contains w^*. Again, the middle string consists of two convex chains linked by a tangent edge, which is available to us due to the preprocessing. We call the oracle on the two vertices of the tangent edge to determine which convex chain contains w^* (or whether the tangent edge contains w^*). After that, since a convex chain is represented by a binary tree of height $O(\log n)$, we can finally locate w^* by calling the oracle on $O(\log n)$ vertices using the binary tree. In this way, after $O(\log n)$ oracle calls, we can reach a fundament string and then w^* can be finally located after another $O(\log n)$ oracle calls. In summary, by calling the oracle on $O(\log n)$ vertices of $\pi(p, q)$, w^* can be found.

To compute the middle point of $\pi(p, q)$, we first compute the geodesic distance $d(p, q)$ in $O(\log n)$ time by using the GH data structure. Then, we can follow the above binary search scheme and each time when the oracle is called on a vertex v, we also keep track of the geodesic distance $d(p, v)$ using the GH data structure. By comparing $d(p, v)$ with $d(p, q)/2$, we can decide which way to proceed the search. In this way, the middle point of $\pi(p, q)$ can be determined in $O(\log n)$ time. □

With Lemma 4.1 at hand, we can find c on $\pi(s, t)$ in $O(\log n)$ time. We can determine whether c is in the interior of $\triangle(s, t, r)$ in $O(\log n)$ time using Lemma 4.1 as follows. First, we determine whether c is the middle point of $\pi(s, t)$. To do so, we first compute the middle point p_{st} of $\pi(s, t)$ by Lemma 4.1. Then, we compute $d(s, p_{st})$ and $d(r, p_{st})$ in $O(\log n)$ time using the GH data structure. It is not difficult to see that p_{st} is c if and only if $d(s, p_{st}) \geq d(r, p_{st})$. If $p_{st} \neq c$, then we use the same way to determine whether the middle point of $\pi(s, r)$ (resp., $\pi(r, t)$) is c. If the above algorithm fails to locate c, then we know that c is in the interior of $\triangle(s, t, r)$.

The above finds c in $O(\log n)$ time for the case where c is not in the interior of $\triangle(s, t, r)$.

4.1.2 c is in the Interior of $\triangle(s, t, r)$

We proceed to the case where c is in the interior of $\triangle(s, t, r)$. Our algorithm utilizes the tentative prune-and-search technique of Kirkpatrick and Snoeyink [17]. First observe that in this case c must be equidistant from all three points s, t, and r. Let s' be the junction vertex of $\pi(s, t)$ and $\pi(s, r)$. Define t' and r' similarly. With the GH data structure, each junction vertex can be computed in $O(\log n)$ time [13]. Define p_s, p_t, and p_r to be the anchors of c in $\pi(s, c)$, $\pi(t, c)$, and $\pi(r, c)$, respectively. Note that the segment connecting c to p_s (resp., p_t, p_r) is tangent to the side of $\triangle(s, t, r)$ that contains it. As c is equidistant to s, t, and r, c is the common intersection of the three bisectors $B(s, t)$, $B(s, r)$, and $B(t, r)$.

Observation 4.2 *The middle point p_{st} of $\pi(s, t)$ must be in $\pi(s', t')$; the middle point p_{sr} of $\pi(s, r)$ must be in $\pi(s', r')$; the middle point p_{tr} of $\pi(t, r)$ must be in $\pi(t', r')$.*

Proof We only prove the case for p_{st} since the other two cases are similar. Assume to the contrary that $p_{st} \notin \pi(s', t')$. Then, either $p_{st} \in \pi(s, s') \setminus \{s'\}$ or $p_{st} \in \pi(t, t') \setminus \{t'\}$. We assume it is the former case as the analysis for the latter case is similar. Then, $d(s, s') > d(t, s')$. Note that $d(s, c) = d(s, s') + d(s', c)$ as c is in the interior of $\triangle(s, t, r)$. Hence, $d(s, c) > d(t, s') + d(s', c) \geq d(t, c)$. But this incurs a contradiction as c is equidistant from s and t. □

Since p_s is the anchor of c in $\pi(s, c)$, p_s has smaller geodesic distance from s than from t or r. Hence, by Observation 4.2, p_s must be in $\gamma_s = \pi(s', p_{st}) \cup \pi(s', p_{sr})$, which consists of two convex chains; we call γ_s a *pseudo-convex chain*. Similarly, p_t must be in $\gamma_t = \pi(t', p_{st}) \cup \pi(t', p_{tr})$ and p_r must be in $\gamma_r = \pi(r', p_{tr}) \cup \pi(r', p_{sr})$. We consider p_{st} and p_{sr} as two ends of γ_s. If we move a point p on γ_s from one end to the other, then the slope of the tangent line of γ_s at p continuously changes. Further, p_s is the only point on γ_s such that $\overline{cp_s}$ is tangent to γ_s. Similar properties hold for γ_t, p_t, γ_r, and p_r.

With the above discussion, we are now in a position to describe our algorithm for computing c. First, we compute the three junction vertices and the three middle points s', t', r', p_{st}, p_{sr}, and p_{tr}. This can be done in $O(\log n)$ time using the GH data structure. To compute c (as well as locate p_s, p_t, and p_r), we resort to the tentative prune-and-search technique [17], as follows.

To avoid the lengthy background explanation, we follow the notation in [17] without definition. We will rely on [17, Theorem 3.9]. To this end, we need to define three continuous and monotone-decreasing functions f, g, and h. We define them in a way similar in spirit to [17, Theorem 4.10] for finding a point equidistant to three convex polygons. Indeed, our problem may be considered as a weighted case of their problem because each point in our pseudo-convex chains has a weight that is equal to its geodesic distance from one of s, t, and r.

We parameterize over [0, 1] each of the three pseudo-convex chains $A = \gamma_s$, $B = \gamma_t$, and $C = \gamma_r$ from one end to the other in counterclockwise order around $\triangle(s, t, r)$. For example, without loss of generality, we assume that s', t', and r' are counterclockwise around $\triangle(s, t, r)$. Then, γ_s is parameterized from p_{sr} to p_{st} over [0, 1], i.e., each value of [0, 1] corresponds to a slope of a tangent at a point on γ_s. For each point a of A, we define $f(a)$ to be the parameter of the point $b \in B$ such that the tangent of A at a and the tangent of B at b intersect at a point on the bisector $B(s, t)$ of s and t (e.g., see Fig. 5). Similarly, we define $g(b)$ for $b \in B$ with respect to C and define $h(c)$ for $c \in C$ with respect to A. One can verify that all three functions are continuous and monotone-decreasing (the tangent at an apex of $\triangle(s, t, r)$ is not unique but the issue can be handled [17]). The fixed-point of the composition of the three functions $h \cdot g \cdot f$ corresponds to c, which can be computed by applying the tentative prune-and-search algorithm of [17, Theorem 3.9].

To see that the algorithm can be implemented in $O(\log n)$ time, we need to show that given any $a \in A$ and any $b \in B$, we can determine whether $f(a) > b$ in $O(1)$ time. To this end, we first find the intersection p of the tangent of A at a and the tangent of B at b. Then, $d(s, a) + |\overline{pa}| < d(t, b) + |\overline{pb}|$ if and only if $f(a) > b$. We will discuss below that the values $d(s, a)$ and $d(t, b)$ will be available during the tentative prune-and-search algorithm. Note that here the tangent of A at a actually refers to the half-line of the tangent whose concatenation with $\pi(s', a)$ is still a convex chain (so that the shortest path can follow that half-line), as shown in Fig. 5. Hence, it is possible that the tangent half-line of a does not intersect the tangent half-line of b. If that happens, either the tangent half-line of a intersects the backward extension of the tangent half-line of b or the backward extension of the tangent half-line of a intersects the tangent half-line of b; in the former case we have $f(a) < b$ and in the latter case $f(a) > b$. Similar properties hold for functions g and h. Finally, we show that we have

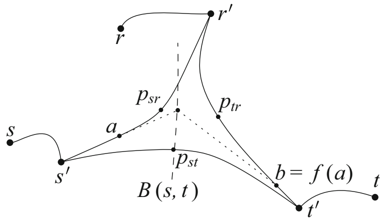

Fig. 5 Illustrating the geodesic triangle $\triangle(s, t, r)$ and the definition of the function $f(a)$ for $a \in A = \gamma_s$

appropriate data structures to represent the three pseudo-convex chains A, B, and C so that the algorithm can terminate in $O(\log n)$ rounds. We only discuss A since the other two cases can be handled similarly. When the algorithm picks the first vertex of A to test, we will use the vertex s'. After the test, the algorithm will proceed on A on one side of s', say, on $\pi(s', p_{st})$. We apply the binary search scheme of Lemma 4.1 on $\pi(s', p_{st})$, which will test $O(\log n)$ vertices. Further, whenever a vertex $a \in \pi(s', p_{st})$ is tested, the binary search scheme of Lemma 4.1 can keep track of $d(s, a)$. Therefore, by applying the tentative prune-and-search technique in [17, Theorem 3.9], we can compute the geodesic center c in $O(\log n)$ time.

The following lemma summarizes our result on the triple-point geodesic center query problem.

Lemma 4.3 *With $O(n)$ time preprocessing, the geodesic center of any three query points in P can be computed in $O(\log n)$ time.*

4.2 Proving Lemma 3.1

With Lemma 4.3, we are ready to present our algorithm for Lemma 3.1. Consider any three sites s, t, r as specified in the statement of Lemma 3.1. Our goal is to compute the point $\alpha(s, t, r)$, which is equidistant from the three sites. Recall that we have two points $\beta(s, t)$ and $\beta(t, r)$ available to us, which are critical to the success of our approach.

The first step of our algorithm is to apply the algorithm for Lemma 4.3 to compute the geodesic center c of the three sites. We check whether c is equidistant to the three sites, in $O(\log n)$ time. If yes, $\alpha(s, t, r) = c$ and we are done. In the following we assume otherwise.

Our algorithm may not compute $\alpha(s, t, r)$ even if it exists, but will guarantee to do so if $\alpha(s, t, r)$ is a Voronoi vertex of FVD(S). This is sufficient for constructing FVD(S) correctly. Hence, in what follows we assume that $\alpha(s, t, r)$ is a Voronoi vertex. This implies that there are two Voronoi edges connecting $\alpha(s, t, r)$ with $\beta(s, t)$ and $\beta(t, r)$, respectively. Recall that c^* is the geodesic center of S. The following observation was discovered by Aronov et al. [3].

Observation 4.4 *If a point p moves from $\beta(s, t)$ (resp., $\beta(t, r)$) to $\alpha(s, t, r)$ along the Voronoi edge, both $d(c^*, p)$ and $d(t, p)$ are monotonically decreasing.*

To simplify the notation, unless otherwise stated, we use α to refer to $\alpha(s,t,r)$. We define the three junction vertices s', t', and r' in the same way as before, which are the three apexes of the geodesic convex hull $\triangle(s,t,r)$. Without loss of generality, we assume that s', t', and r' are counterclockwise around the boundary of $\triangle(s,t,r)$ (e.g., see Fig. 4). We define p_s, p_t, and p_r as the anchors of α in $\pi(s,\alpha)$, $\pi(t,\alpha)$, and $\pi(r,\alpha)$, respectively. Define p_{st}, p_{sr}, and p_{tr} as the middle points of $\pi(s,t)$, $\pi(s,r)$, and $\pi(t,r)$, respectively. The following observation, obtained from the results of Aronov [2], will occasionally be used later.

Observation 4.5 *Suppose x and y are two points of P such that their bisector $B(x,y)$ does not contain any vertex of P. Then, for any point $z \in P$, the shortest path $\pi(x,z)$ (resp., $\pi(y,z)$) either does not intersect $B(x,y)$ or intersects it at a single point.*

Proof All arguments here are from Aronov [2]. $B(x,y)$ divides P into two subpolygons; one of them, denoted by P_x, contains x and the other, denoted by P_y, contains y. We assume that neither P_x nor P_y contains $B(x,y)$. All points in P_x are closer to x than to y and all points in P_y are closer to y than to x. Let z be any point in P. If $z \in P_x$, then $\pi(x,z)$ is in P_x. If $z \in B(x,y)$, then $\pi(x,z) \setminus \{z\}$ is in P_x due to the general position assumption. If $z \in P_y$, then $\pi(x,z)$ intersects $B(x,y)$ at a single point. Similar results hold for $\pi(y,z)$. $\square$

Recall that $\alpha(s,t,r)$ is not c. Hence, c must be equidistant to two sites and the geodesic distance from them to c is strictly larger than that from the third site to c. Depending on what the two sites are, there are three cases $d(c,s) = d(c,r) > d(c,t)$, $d(c,t) = d(c,r) > d(c,s)$, and $d(c,t) = d(c,s) > d(c,r)$. The following lemma shows that the latter two cases cannot happen.

Lemma 4.6 *If $\alpha(s,t,r)$ is a Voronoi vertex of* FVD(S)*, then neither $d(c,t) = d(c,r) > d(c,s)$ nor $d(c,t) = d(c,s) > d(c,r)$ can happen.*

Proof Note that the two cases are symmetric and thus we only discuss the case $d(c,t) = d(c,r) > d(c,s)$. Assume to the contrary that $d(c,t) = d(c,r) > d(c,s)$ happens. Then, c is the middle point of $\pi(t,r)$. Consider the farthest Voronoi diagram FVD(s,t,r) with respect to the three sites s,t,r only (without considering other sites of S). Then, α is a vertex of the diagram, i.e., the three Voronoi edges bounding the three cells of s, t, and r meet at α (e.g., see Fig. 6). Since $d(c,t) = d(c,r) > d(c,s)$, c is on the Voronoi edge $E(t,r)$ bounding the cells of t and r. By the definition of $\beta(t,r)$, it is also on $E(t,r)$. Hence, all three points α, c, and $\beta(t,r)$ are on $E(t,r)$. Let $E'(t,r)$ be the portion of $E(t,r)$ between α and $\beta(t,r)$. Since both $\beta(t,r)$ and α are vertices of FVD(S), $E'(t,r)$ must be an edge of FVD(S) bounding the two cells of t and r. By Observation 4.4, if we move a point p on $E'(t,r)$ from $\beta(t,r)$ to α, $d(t,p)$ is monotonically decreasing.

Since c is the middle point of $\pi(t,r)$, if we move a point p on the bisector $B(t,r)$ from one end to the other, $d(t,p)$ will first strictly decreases until $p = c$ and then strictly increases. Note that $E'(t,r) \subseteq E(t,r) \subseteq B(t,r)$. Note also that $E'(t,r)$ has α and $\beta(t,r)$ as its two endpoints. Depending on whether $E'(t,r)$ contains c, there are two cases.

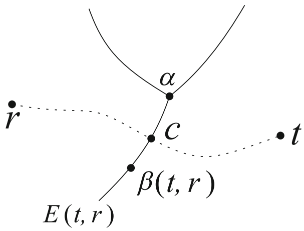

Fig. 6 Illustrating the proof of Lemma 4.6. The three solid curves are the three Voronoi edges of FVD(s, t, r). The dotted curve is the shortest path $\pi(r, t)$. The curve between α and $\beta(t, r)$ is $E'(t, r)$

- If $E'(t, r)$ does not contain c, then c, $\beta(t, r)$, and α appear in $E(t, r)$ in this order since α is a vertex of FVD(s, t, r) and thus is an endpoint of $E(t, r)$. Hence, c, $\beta(t, r)$, and α appear in $B(t, r)$ in this order. Therefore, if we move a point p on $E'(t, r)$ from $\beta(t, r)$ to α, $d(t, p)$ must be strictly increasing. But this contradicts the fact that if we move a point p on $E'(t, r)$ from $\beta(t, r)$ to α, $d(t, p)$ is monotonically decreasing.
- If $E'(t, r)$ contains c, then $\beta(t, r)$, c, and α appear in $E(t, r)$ in this order (e.g., see Fig. 6). Hence, $\beta(t, r)$, c, and α appear in $B(t, r)$ in this order. Therefore, if we move a point p on $E'(t, r)$ from $\beta(t, r)$ to α, $d(t, p)$ will first strictly decrease and then strictly increase, a contradiction again.

The lemma thus follows. □

Note that Lemma 4.6 is obtained based on the assumption that $\alpha(s, t, r)$ is a Voronoi vertex of FVD(S). Therefore, if one of the two cases in Lemma 4.6 happens during the algorithm, then we can simply return null.

In what follows, we assume that $d(c, s) = d(c, r) > d(c, t)$. Thus, c must be the middle point of $\pi(s, r)$. Depending on the location of c, there are three cases: $c \in \pi(s', r')$, $c \in \pi(s, s') \setminus \{s'\}$, and $c \in \pi(r', r) \setminus \{r'\}$. The latter two cases are symmetric, so we will only discuss the first two cases.

4.2.1 The Case $c \in \pi(s', r')$

Let $\overline{uv}$ be the edge of $\pi(s', r')$ containing c such that $d(s, u) < d(s, v)$. It is possible that u is s or/and v is t. We first assume that both u and v are polygon vertices; we will show later the other case (i.e., at least one of u and v is not a polygon vertex) can be reduced to this case. Our algorithm relies on the following lemma (e.g., see Fig. 7).

Lemma 4.7 (i) *α must be in the geodesic triangle $\triangle(s, r, \beta(s, t))$.*

(ii) *The apexes of $\triangle(s, r, \beta(s, t))$ are u', v', and $\beta(s, t)$, where u' (resp., v') is the junction vertex of and $\pi(s, r)$ and $\pi(s, \beta(s, t))$ (resp., $\pi(r, \beta(s, t))$) (in Fig. 7, $u' = u$ and $v' = v$).*

(iii) *p_s must be on the pseudo-convex chain $\pi(u', \beta(s, t)) \cup \pi(u', v')$ and $\overline{\alpha p_s}$ is tangent to the chain.*

(iv) *p_r must be on the pseudo-convex chain $\pi(v', \beta(s, t)) \cup \pi(v', u')$ and $\overline{\alpha p_r}$ is tangent to the chain.*

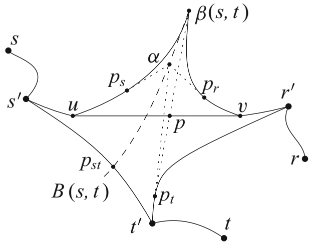

Fig. 7 Illustrating Lemma 4.7. In this example, $u' = u$ and $v' = v$

(v) *p_t must be on the pseudo-convex chain $\pi(t_{uv}, u) \cup \pi(t_{uv}, v)$ and $\overline{\alpha p_t}$ is tangent to the chain, where t_{uv} is the junction vertex of $\pi(t, u)$ and $\pi(t, v)$.*

(vi) *$\overline{\alpha p_t}$ intersects $\overline{uv}$.*

Proof As both u and v are polygon vertices, $\overline{uv}$ divides P into two sub-polygons; one of them, denoted by P_1, does not contain t and we use P_2 to denote the other one. Oh and Ahn [22] claimed without proof (in the proof of [22, Lemma 3.6]) that α is in P_1. We provide a brief proof below.

Assume to the contrary that α is not in P_1. Then, $\alpha \in P_2$. Since $B(s, r)$ intersects $\pi(s, r)$ only once at $c \in \overline{uv} \subseteq \pi(s, r)$, $B(s, r)$ is partitioned into two portions by c, one in P_1 and the other in P_2; let $B_2(s, r)$ denote the portion in P_2, which has c as one of its endpoint. As α is equidistant from s, t, and r, α is on $B(s, t)$. Since $\alpha \in P_2$, we obtain that $\alpha \in B_2(s, r)$. As α is not c, which is the geodesic center of s, t, and r, α cannot be in the geodesic triangle $\triangle(s, t, r)$. Therefore, if we move on $B_2(s, r)$ from c to its other endpoint, we will first enter $\triangle(s, t, r)$ and then encounter either $\pi(s', t')$ or $\pi(t', r')$ before we encounter α. Without loss of generality, we assume that we encounter $\pi(s', t')$. We assume that s, t, r are ordered counterclockwise around the boundary of their geodesic hull (e.g., see Fig. 7). Consider the farthest Voronoi diagram FVD(s, t, r) of the three sites s, t, r only (without considering other sites of S). Let $C(p)$ be the cell of $p \in \{s, t, r\}$ in the diagram. As $d(c, s) = d(c, r) > d(c, t)$, c belongs to the common boundary of $C(s)$ and $C(r)$, i.e., c is on an edge of FVD(s, t, r). The point α divides $B(s, r)$ into two portions, one of which contains c. The above implies that the portion of $B(s, r)$ containing c is an edge of FVD(s, t, r) (e.g., see Fig. 8). That edge partitions $\triangle(s, t, r)$ into two sides; one side contains s' and the other contains r'. It is not difficult to see that the side containing s' belongs to $C(r)$ while the other side belongs to $C(s)$. Then, one can verify that the three cells $C(s)$, $C(t)$, and $C(r)$ in FVD(s, t, r) are ordered clockwise along the boundary of P (e.g., see Fig. 8). According to Aronov [3], s, t, and r should also be ordered clockwise around the boundary of their geodesic hull. But this contradicts the fact that s, t, and r are ordered counterclockwise around the boundary of their geodesic hull.

The above proves that α is in P_1. Oh and Ahn [22] showed that $\overline{\alpha p_t}$ intersects $\overline{uv}$. The main idea of the proof is that if this were not the case, then a vertex of P would be on a bisector of two of the three sites s, t, and r, contradicting with the general position assumption (see the proof of [22, Lemma 3.6] for the detailed analysis). This leads to the lemma statement (vi), which further implies the lemma statement (v).

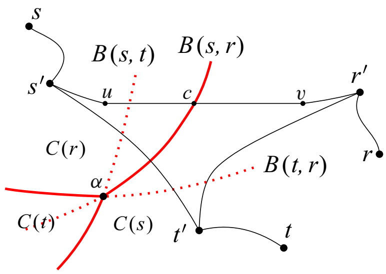

Fig. 8 Illustrating FVD(s,t,r), whose edges are depicted by thick (red) solid curves. The (red) dotted curves belong to bisectors but not on FVD(s,t,r). The point α is the only vertex of FVD(s,t,r) because it is equidistant from all three sites. The cells $C(s)$, $C(t)$, and $C(r)$ are ordered clockwise around α, while s, t, and r are ordered counterclockwise around the boundary of their geodesic hull, a contradiction

To simplify the notation, let $q = \beta(s,t)$. By definition, q is on the bisector $B(s,t)$. As α is equidistant from s, t, and r, α is also on $B(s,t)$. Let p_{st} be the middle point of $\pi(s,t)$. Hence, $p_{st} \in B(s,t)$.

We claim that α must be on $B(s,t)$ between p_{st} and q (e.g., see Fig. 9). Indeed, notice that p_{st} is the point on $B(s,t)$ closest to t and if we move a point p from one end of $B(s,t)$ to the other end, $d(t,p)$ will first monotonically decrease until p_{st} and then monotonically increase. By Observation 4.4, if we move a point p along $B(s,t)$ from q to α, $d(t,p)$ will monotonically decrease. As such, α must be on $B(s,t)$ between p_{st} and q.

We next argue that $q \in P_1$. Depending on whether $B(s,t)$ intersects $\overline{uv}$, there are two cases. If $B(s,t)$ does not intersect $\overline{uv}$, then as $\alpha \in B(s,t)$ and $\alpha \in P_1$, $B(s,t)$ is in P_1. Since $q \in B(s,t)$, $q \in P_1$ holds. If $B(s,t)$ intersects $\overline{uv}$, then since $\overline{uv} \subseteq \pi(s,r)$, by Observation 4.5, $B(s,t)$ intersects $\overline{uv}$ at a single point, denoted by q_{st} (e.g., see Fig. 9). To prove $q \in P_1$, since $\alpha \in P_1$ and α is on $B(s,t)$ between q and p_{st}, it suffices to show that $p_{st} \in P_2$. Indeed, since $B(s,t)$ intersects $\overline{uv}$ at q_{st} and $\overline{uv} \subseteq \pi(s,r)$, by Observation 4.5, $B(s,t)$ does not intersect any other point of $\pi(s,r)$. Hence, $B(s,t)$ does not intersect $\pi(s,s') \setminus \{s'\}$, which is a subpath of $\pi(s,r)$ and does not contain any point of $\overline{uv}$. This implies that p_{st} cannot be on $\pi(s,s') \setminus \{s'\}$ and thus is on $\pi(s',t)$. Note that s' is in P_2. Since both s' and t are in P_2, $\pi(s',t)$ is in P_2. As such, $p_{st} \in P_2$.

The above proves that $q \in P_1$ (recall $q = \beta(s,t)$) and α is on $B(s,t)$ between q and p_{st}. In the following, we proceed to prove that $\alpha \in \triangle(s,r,q)$.

As α is on $B(s,t)$ between q and p_{st}, α must be in the geodesic triangle $\triangle(s,t,q)$. Since $q \in B(s,t)$, due to the general position assumption, the incident edges of q in $\pi(q,s)$ and $\pi(q,t)$ cannot be coincident [3]. This means that q is the junction vertex of $\pi(q,s)$ and $\pi(q,t)$, and thus q is an apex of $\triangle(s,t,q)$. Since $t \in P_2$ and $q \in P_1$, $\pi(q,t)$ must cross $\overline{uv}$ at a point p; e.g., see Fig. 7. Since both α and q are in P_1, α is also in the geodesic triangle $\triangle(s,p,q)$ and q is an apex of $\triangle(s,p,q)$. Further, since $p \in \overline{uv} \subseteq \pi(s,r)$, $\triangle(s,p,q)$ is a subset of $\triangle(s,r,q)$ and q is also an apex of $\triangle(s,r,q)$. As such, we obtain that $\alpha \in \triangle(s,r,q)$. This proves the lemma statement (i).

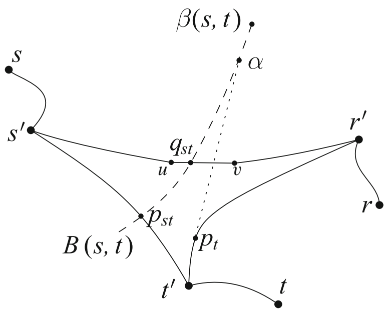

Fig. 9 Illustrating the relative positions of $\beta(s,t)$, α, and p_{st}

The above also proves that q is an apex of $\triangle(s,r,q)$. By definition, u' and v' are the other two apexes of $\triangle(s,r,q)$. This proves the lemma statement (ii). Since $\alpha \in \triangle(s,r,q)$, the lemma statements (iii) and (iv) obviously hold. □

In light of Lemma 4.7, we can apply the tentative prune-and-search technique [17] on the three pseudo-convex chains specified in the lemma in a similar way as before to compute α in $O(\log n)$ time.

We summarize our algorithm for this case. First, we compute the edge $\overline{uv}$, which can be done in $O(\log n)$ time using the GH data structure by Lemma 4.1. Second, we compute the junction vertex t_{uv} of $\pi(t,u)$ and $\pi(t,v)$ in $O(\log n)$ time by the GH data structure [13]. Third, we apply the tentative prune-and-search technique on the three pseudo-convex chains as specified in Lemma 4.7, along with the binary search scheme in Lemma 4.1 on the chains, to compute α in $O(\log n)$ time.

Recall that the above algorithm is based on the assumption that α is a Voronoi vertex of FVD(S). However, when we invoke the procedure during the geodesic sweeping algorithm we do not know whether the assumption is true. Therefore, as a final step, we add a validation procedure as follows. Suppose α is the point returned by the algorithm. First, we check whether $d(s,\alpha) = d(t,\alpha) = d(r,\alpha)$. If not, we return null. Otherwise, we further check whether $d(c^*,\alpha) \le \min\{d(c^*,\beta(s,t)), d(c^*,\beta(t,r))\}$. This is because the Voronoi vertex α is only useful if it is inside the current sweeping circle C, whose geodesic distance to c^* is at most $\min\{d(c^*,\beta(s,t)), d(c^*,\beta(t,r))\}$ (because neither $\beta(s,t)$ nor $\beta(t,r)$ is in the interior of C). Hence, if $d(c^*,\alpha) \le \min\{d(c^*,\beta(s,t)), d(c^*,\beta(t,r))\}$, then we return α; otherwise, we return null. This validation step takes $O(\log n)$ time by the GH data structure.

At least one of u and v is not a polygon vertex. The above discusses the case where both u and v are polygon vertices. In the following, we consider the other case where at least one of them is not a polygon vertex, i.e., $u = s$ or/and $v = r$ (because all vertices of $\pi(s,r)$ except s and r are polygon vertices). In fact, this case is missed from the algorithm of Oh and Ahn [22] (see the proof of [22, Lemma 3.6]). It turns out that Lemma 4.7 still holds for this case and thus we can apply exactly the same algorithm as above. We prove the lemma below by reducing this case to the previous case where u and v are polygon vertices.

Lemma 4.8 *Lemma 4.7 still holds when $u = s$ or/and $v = r$.*

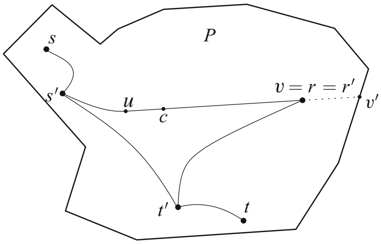

Fig. 10 Illustrating the proof of Lemma 4.8

Proof Without loss of generality, we assume that v is not a polygon vertex and thus $v = r = r'$. We extend $\overline{uv}$ in the direction from u to v until ∂P at a point v' (e.g., see Fig. 10). If u is also not a polygon vertex, then $u = s = s'$ and we extend $\overline{uv}$ in the direction from v to u until ∂P at a point u'. If u is a polygon vertex, we let $u' = u$.

Let P' be the polygon by merging P with the two segments $\overline{vv'}$ and $\overline{uu'}$. So P' is a (weakly) simple polygon with u and v as two vertices. We claim that $B(s,t)$, $B(s,r)$, and $B(t,r)$ are still the bisectors of s, t, and r in P'. Before proving the claim, we proceed to prove the lemma with help of the claim. Due to the claim, since u and v are now both polygon vertices of P', we can apply literally the same argument as in the previous case. Indeed, the argument only relies on the properties of the three bisectors, e.g., α is their common intersection. Now that the three bisectors do not change from P to P', the same argument still works. Thus, the lemma follows.

In the following, we prove the above claim. It is sufficient to show the following four properties:

(a) $\overline{vv'} \setminus \{v\} \cup \overline{uu'} \setminus \{u\}$ does not intersect any of the three bisectors $B(s,t)$, $B(t,r)$, and $B(s,r)$;
(b) $\overline{vv'} \setminus \{v\} \cup \overline{uu'} \setminus \{u\}$ does not intersect $\pi(s,p)$ for any point $p \in B(s,t) \cup B(s,r)$;
(c) $\overline{vv'} \setminus \{v\} \cup \overline{uu'} \setminus \{u\}$ does not intersect $\pi(t,p)$ for any point $p \in B(s,t) \cup B(t,r)$;
(d) $\overline{vv'} \setminus \{v\} \cup \overline{uu'} \setminus \{u\}$ does not intersect $\pi(r,p)$ for any point $p \in B(s,r) \cup B(t,r)$.

Below we will prove the above four properties only for $\overline{vv'} \setminus \{v\}$, as the proof for $\overline{uu'} \setminus \{u\}$ is similar. We prove these properties in order.

Property (a). First of all, since v' is an extension of $\overline{uv}$, it holds that $\pi(s,v') = \pi(s,v) \cup \overline{vv'}$. Recall that $d(s,c) = d(c,r) > d(t,c)$ and $c \in \overline{uv}$. Assume to the contrary that $\overline{vv'}$ intersects $B(s,t)$, say, at a point z. Then, $d(s,z) = d(s,c) + |\overline{cz}|$. On the other hand, by triangle inequality, $d(t,z) \le d(t,c) + |\overline{cz}|$. Hence, we obtain $d(s,z) = d(s,c) + |\overline{cz}| > d(t,c) + |\overline{cz}| \ge d(t,z)$. However, since $z \in B(s,t)$, $d(s,z) = d(t,z)$, and thus contradiction occurs. This proves that $\overline{vv'}$ does not intersect $B(s,t)$. Because $\pi(s,v')$ contains $\overline{rv'}$, $d(s,p) > d(r,p)$ for any point $p \in \overline{vv'}$. Therefore, $\overline{vv'} \setminus \{v\}$ cannot intersect $B(s,r)$.

Next we prove the case for $B(t,r)$. Let a be the anchor of r in $\pi(t,r)$. Since $d(t,c) < d(r,c) = |\overline{cr}|$, the angle $\angle(a,r,c)$ must be smaller than $\pi/2$, and thus the angle $\angle(v',r,a)$ is larger than $\pi/2$. Notice that r is the junction vertex of $\pi(r,t)$ and $\pi(r,p) = \overline{rp}$ for any $p \in \overline{vv'} \setminus \{v\}$. Since the angle $\angle(p,r,a) = \angle(v',r,a)$ is

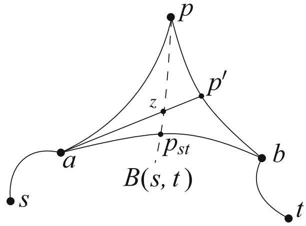

Fig. 11 Illustrating the pseudo-triangle $\triangle(s, t, p)$

larger than $\pi/2$, it must hold that $d(t, p) > d(r, p)$ (see [25, Cor. 2]). This implies that p cannot be on $B(t, r)$. Thus, $\overline{vv'} \setminus \{v\}$ does not intersect $B(t, r)$. This proves property (a).

Property (b). Let p be any point in $B(s, t) \cup B(s, r)$. Assume to the contrary that $\pi(s, p)$ contains a point $p' \in \overline{vv'} \setminus v$. Then, since $\pi(s, p')$ contains $\pi(s, r)$, $\pi(s, p)$ contains $\pi(s, r)$ and thus contains r.

If $p \in B(s, r)$, then we immediately obtain contradiction as $\pi(s, p)$ cannot contain r. Now consider the case $p \in B(s, t)$. Since $d(s, c) > d(t, c)$, there must be a point $p'' \in \pi(s, c)$ such that $d(s, p'') = d(t, p'')$, i.e., $p'' \in B(s, t)$. Since $\pi(s, p'') \subseteq \pi(s, c)$ and $c \neq r$, $\pi(s, p'')$ does not contain r. If $p = p''$, we obtain that $\pi(s, p)$ does not contain r, which incurs a contradiction. Hence, $p \neq p''$. Thus, $\pi(s, p)$ intersects $B(s, t)$ at two different points p and p''. But this is not possible due to Observation 4.5. This proves property (b).

Property (c). For (c), let p be any point in $B(s, t) \cup B(t, r)$. Assume to the contrary that $\pi(t, p)$ contains a point $p' \in \overline{vv'} \setminus \{v\}$. We first discuss the case $p \in B(s, t)$. Consider the geodesic triangle $\triangle(s, t, p)$; e.g., see Fig. 11. Since $p \in B(s, t)$, p must be an apex of $\triangle(s, t, p)$. Let a be the junction vertex of $\pi(s, p)$ and $\pi(s, t)$ and let b the junction vertex of $\pi(t, s)$ and $\pi(t, p)$. Hence, a and b are two apexes of $\triangle(s, t, p)$. By a similar argument as Observation 4.2, the middle point p_{st} of $\pi(s, t)$ must be on $\pi(a, b)$, i.e., the side of $\triangle(s, t, p)$ opposite to p. Hence, the portion of $B(s, t)$ between p and p_{st}, denoted by B, separates $\triangle(s, t, p)$ into two parts. As $\pi(t, p) = \pi(t, b) \cup \pi(b, p)$, p' is either in $\pi(t, b)$ or in $\pi(b, p)$.

- If p' is in $\pi(t, b)$, then p' is in $\pi(s, t)$ as $\pi(t, b)$ is a subpath of $\pi(s, t)$. Therefore, $\pi(s, p')$ is a subpath of $\pi(s, t)$. Recall that $\pi(s, r) \subseteq \pi(s, p')$. We thus obtain that $\pi(s, r)$ is a subpath of $\pi(s, t)$. Since $c \in \pi(s, r)$, we obtain that s, c, r, and t are all on $\pi(s, t)$ in this order. Hence, $d(c, r) \leq d(c, t)$, which incurs a contradiction as $d(c, r) > d(c, t)$.
- If p' is in $\pi(b, p)$, then $\pi(s, p')$ must intersect B, say, at a point z (e.g., see Fig. 11). By Lemma 4.9, it holds that $d(s, z) \geq d(z, p')$, and thus $d(s, z) \geq d(s, p')/2$. Since $r \in \pi(s, p')$, we have $d(s, z) \geq d(s, r)/2 = d(s, c)$. This implies that $c \in \pi(s, z)$. Because $z \in B(s, t)$, we have $d(s, c) \leq d(t, c)$. But this contradicts fact $d(s, c) > d(t, c)$.

The above obtains contradiction for the case $p \in B(s, t)$.

We next discuss the case $p \in B(t, r)$. The bisector $B(t, r)$ divides P into two subpolygons; let P_t be the one containing t and let P_r denote the one containing r. We assume that neither P_s nor P_r contains $B(t, r)$. As $p \in B(t, r)$, the entire path $\pi(t, p)$ is in $P_t \cup B(s, t)$. Since $p' \in \overline{vv'} \setminus \{v'\}$, by a similar argument using the angle at r as that for property (a), we can show that $d(t, p') > d(r, p')$. This implies that p' is in P_r. Therefore, p' cannot be in $\pi(s, p)$, a contradiction. This proves property (c).

Property (d). For (d), assume to the contrary that $\pi(r, p)$ contains a point $p' \in \overline{vv'} \setminus \{v\}$. We first discuss the case $p \in B(t, r)$. Let a be the anchor of r in $\pi(t, r)$. Recall that we have shown before that the angle $\angle(a, r, v')$ is larger than $\pi/2$. Consider the geodesic triangle $\triangle(t, r, p)$.

- If r is not an apex of $\triangle(t, r, p)$, then $\overline{ra} \in \pi(r, t) \cap \pi(r, p)$. Since $p' \in \pi(r, p)$ and $p' \notin \overline{ra}$, we obtain that $\pi(r, p')$ contains a. However, since $\angle(a, r, v') > \pi/2$ and $p' \in \overline{rv'}$, $\pi(r, p') = \overline{rp'}$ does not contain a, a contradiction.
- If r is an apex of $\triangle(t, r, p)$, then since $p \in B(t, r)$, the angle $\angle(a, r, b)$ must be smaller than $\pi/2$, where b is the anchor of r in $\pi(p, r)$. As $p' \in \pi(r, p)$ and $\pi(r, p') = \overline{rp'}$, we obtain that $p' \in \overline{rb}$ and thus $\angle(a, r, b) = \angle(a, r, p')$. Hence, $\angle(a, r, p')$ is smaller than $\pi/2$. However, $\angle(a, r, p') = \angle(a, r, v')$, which is larger than $\pi/2$. Thus we obtain a contradiction.

We then discuss the case $p \in B(s, r)$. Note that u is the anchor of r in $\pi(s, r)$. Hence, the angle $\angle(u, r, v')$ is equal to π, which is larger than $\pi/2$. Consequently, we can follow the same analysis as above to obtain contradiction. This proves property (d). The lemma thus follows. □

We finally prove the following technical lemma, which is needed in the proof of Lemma 4.8. The lemma, which establishes a very basic property of shortest paths in simple polygons, may be interesting in its own right.

Lemma 4.9 *Let s and t be any two points in P such that $B(s, t)$ does not contain any vertex of P. Suppose p is a point in $B(s, t)$ and p' is a point in $\pi(t, p)$. Then, $\pi(s, p')$ intersects $B(s, t)$ at a single point z and $d(s, z) \geq d(z, p')$ (in particular, $d(s, z) > d(z, p')$ if $p' \neq t$); e.g., see Fig. 11.*

Proof We first consider a special case where p is the middle point p_{st} of $\pi(s, t)$. Note that $p_{st} \in B(s, t)$. In this case, $z = p_{st}$ and $\pi(s, t) = \pi(s, z) \cup \pi(z, t)$. Hence, $\pi(s, p') = \pi(s, z) \cup \pi(z, p')$ and $d(s, z) = d(z, t) \geq d(z, p')$ (and $d(s, z) = d(s, t) > d(z, p')$ if $p' \neq t$).

In the following we assume $p \neq p_{st}$. Consider the geodesic triangle $\triangle(s, t, p)$. Since $p \in B(s, t)$, p is an apex of $\triangle(s, t, p)$. Let a be the junction vertex of $\pi(s, t)$ and $\pi(s, p)$ and b be the junction vertex of $\pi(t, s)$ and $\pi(t, p)$ (e.g., see Fig. 11). Hence, a, b, p are the three apexes of $\triangle(s, t, p)$. In the following discussion we will use $\triangle(a, b, p)$ instead. By a similar argument as Observation 4.2, p_{st} is in $\pi(a, b)$.

The bisector $B(s, t)$ partitions P into two connected subpolygons P_s and P_t such that P_s contains s and P_t contains t [2]. We assume that neither P_s nor P_t contains $B(s, t)$. Since $p \in B(s, t)$, $\pi(s, p) \setminus \{p\} \subseteq P_s$ and $\pi(t, p) \setminus \{p\} \subseteq P_t$ [2]. This implies that $\pi(s, p')$ intersects $B(s, t)$ at a single point z because $s \in P_s$ and $p' \notin P_s$. Since $p' \in \pi(t, p)$, p' is either in $\pi(t, b)$ or in $\pi(b, p) \setminus \{b\}$. In the former case,

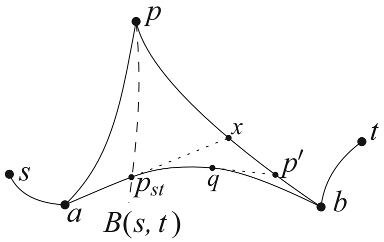

Fig. 12 Illustrating the proof of Lemma 4.9 for the case $q \in \pi(p_{st}, b) \setminus \{p_{st}\}$

$\pi(s, b) \subseteq \pi(s, p') \subseteq \pi(s, t)$ and thus $z = p_{st}$. Hence, $d(s, z) = d(s, t)/2$. On the other hand, $\pi(z, p') \subseteq \pi(z, t)$ and thus $d(z, p') \leq d(z, t) = d(s, t)/2$ (and $d(z, p') < d(z, t)$ if $p' \neq t$). Therefore, we obtain that $d(s, z) \geq d(z, p')$ (and $d(s, z) > d(z, p')$ if $p' \neq t$).

It remains to discuss the case $p' \in \pi(b, p) \setminus \{b\}$. In this case, p' is a point on the boundary of $\triangle(a, b, p)$. Let q be the anchor of p' in $\pi(s, p')$. Because p' is on the boundary of $\triangle(a, b, p)$, $\pi(s, a) \subseteq \pi(s, p')$ and q is on the boundary of $\triangle(a, b, p)$. Since p' is in $\pi(b, p)$, which is the opposite side of the apex a, q cannot be on $\pi(b, p)$ and thus must be on $\pi(a, p) \cup \pi(a, b)$. Note that $\pi(a, b) = \pi(a, p_{st}) \cup \pi(p_{st}, b)$.

Our goal is to prove $d(s, z) > d(z, p')$. Assume to the contrary that $d(s, z) \leq d(z, p')$. In the following we will obtain $d(s, p) > d(t, p)$, which incurs a contradiction as $p \in B(s, t)$. Depending on whether q is in $\pi(a, p) \cup \pi(a, p_{st})$ or in $\pi(p_{st}, b) \setminus \{p_{st}\}$, there are two cases. We first show that the latter case can be reduced to the former case.

If $q \in \pi(p_{st}, b) \setminus \{p_{st}\}$, then $z = p_{st}$; e.g., see Fig. 12. Hence, $d(z, b) \leq d(t, z) = d(s, z) \leq d(z, p')$. If we move a point x on $\pi(b, p)$ from b to p, then $d(s, x)$ is strictly convex (see the proof of [25, Lemma 1]), and more precisely, $d(s, x)$ first strictly decreases until a point x^* and then strictly increases. We claim that $x^* \in \pi(b, p')$. Indeed, assume to the contrary that $x^* \in \pi(p', p)$. Then, we can obtain $d(s, p') < d(s, b)$. Because both $\pi(s, p')$ and $\pi(s, b)$ contain z, we derive that $d(z, p') < d(z, b)$, which contradicts $d(z, p') \geq d(z, b)$. Due to the above claim, as we move x from p' to p along $\pi(p', p)$, $\pi(s, x)$ will strictly increase. We stop moving x when $\overline{xx'}$ contains p_{st} (e.g., see Fig. 12), where x' is the anchor of x in $\pi(s, x)$. Note that such a moment must exist as $\pi(s, p)$ does not contain p_{st}. As $p_{st} \in \pi(s, p')$, z is still p_{st}. Since $d(s, x) > d(s, p')$ and both $\pi(s, x)$ and $\pi(s, p')$ contain z, we can obtain that $d(z, p') < d(z, x)$. As $d(s, z) \leq d(z, p')$, we deduce that $d(s, z) \leq d(z, x)$. Now we obtain an instance of the first case (i.e., $q \in \pi(a, p) \cup \pi(a, p_{st})$) because $x' \in \pi(a, p_{st})$, i.e., we can consider x as a new point p' and obtain contradiction by using the analysis given below for the first case.

In the following we consider the case where $q \in \pi(a, p) \cup \pi(a, p_{st})$; e.g., see Fig. 13. In this case, $\overline{qp'}$ intersects $B(s, t)$ at z and thus $d(z, p') = \overline{zp'}$. There are two subcases depending on whether $q \in \pi(a, p)$ or $q \in \pi(a, p_{st})$. For each case, we will construct a geodesic triangle $\triangle(a', b', p)$ (which may not be in P) with the following properties: (1) a', b', and p are its three apexes; (2) the length of the side of $\triangle(a', b', p)$ connecting a' and p, denoted by $l(a', p)$, is at most $d(s, p)$; (3) the

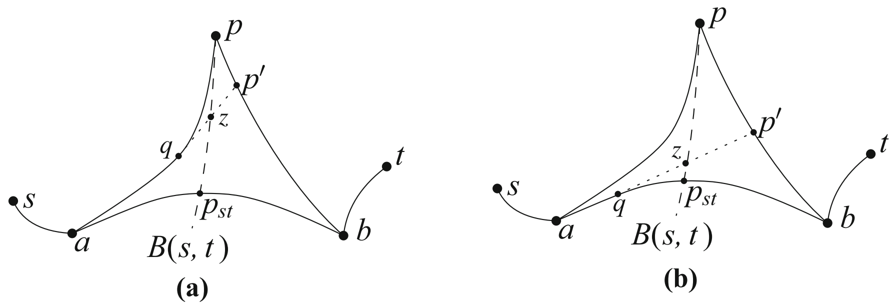

Fig. 13 Illustrating the proof of Lemma 4.9: (**a**) $q \in \pi(a, p)$; (**b**) $q \in \pi(a, p_{st})$

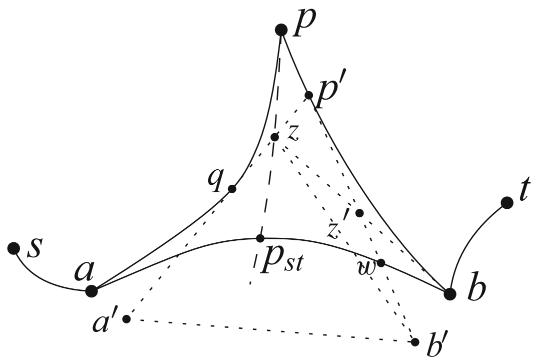

Fig. 14 Illustrating the definitions of a', b', w, and z' in the proof of Lemma 4.9

length of the side of $\triangle(a', b', p)$ connecting b' and p, denoted by $l(b', p)$, is equal to $d(b, p)$; (4) the angle at b' formed by its two incident edges of $\triangle(a', b', p)$ is at least $\pi/2$. These properties together lead to $d(s, p) > d(t, p)$. Indeed, due to property (4), it holds that $l(b', p) < l(a', p)$ (see [25, Cor. 2]). Combining with properties (2) and (3), we have $d(t, p) = l(b', p) < l(a', p) \le d(s, p)$. This incurs a contradiction since $d(t, p) = d(s, p)$.

The first subcase $q \in \pi(a, p)$. We begin with the subcase $q \in \pi(a, p)$; e.g., see Fig. 13(a). We extend $\overline{p'q}$ along the direction from p' to q until a point a' such that $|\overline{qa'}| = d(s, q)$; note that $\overline{qa'}$ may not be in P. Refer to Fig. 14. Note that $\pi(p, q) \cup \overline{qa'}$ is still a convex chain and its length $l(a', p)$ is equal to $|\overline{qa'}| + d(q, p) = d(s, q) + d(q, p) = d(s, p)$. Let p'' be the vertex of $\pi(p', p)$ incident to p'. We extend $\overline{p''p'}$ along the direction from p'' to p' until a point b' such that $|\overline{p'b'}| = d(t, p')$. Note that $\pi(p, p') \cup \overline{p'b'}$ is still a convex chain and its length $l(b', p)$ is equal to $|\overline{p'b'}| + d(p', p) = d(t, p') + d(p', p) = d(t, p)$. In the following we show that the angle $\angle(p', b', a')$ is at least $\pi/2$; this will prove all four properties described above for the geodesic triangle $\triangle(a', b', p)$.

We claim that $|\overline{zb'}| \le d(z, t)$. Before proving the claim, we first show $\angle(p', b', a') \ge \pi/2$ by using the claim. Indeed, notice that $|\overline{za'}| = |\overline{zq}| + |\overline{qa'}| = |\overline{zq}| + d(s, q) = d(s, z)$. As $z \in B(s, t)$, $d(s, z) = d(t, z)$. Hence, $|\overline{za'}| \ge |\overline{zb'}|$. Recall that $d(s, z) \le d(z, p') = |\overline{zp'}|$. Therefore, we have $|\overline{zp'}| \ge |\overline{za'}| \ge |\overline{zb'}|$. If we

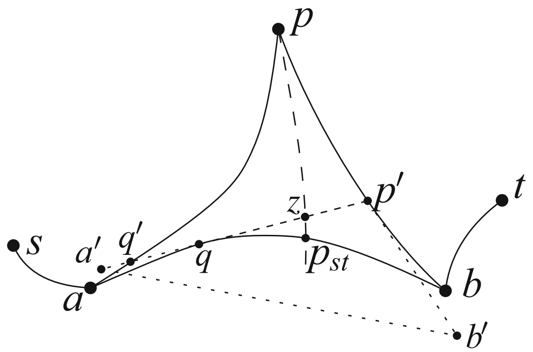

Fig. 15 Illustrating the definitions of a', b', and q' in the proof of Lemma 4.9

draw a circle centered at z with radius equal to $|\overline{za'}|$, then a' is on the circle, b' is inside or on the circle, and p' is outside or on the circle. Since $\overline{a'p'}$ contains a diameter of the circle, we obtain that $\angle(p', b', a') \geq \pi/2$.

We proceed to prove the claim $|\overline{zb'}| \leq d(z, t)$. Notice that $\overline{p'b'}$ must intersect $\pi(z, b)$. Indeed, unless $\overline{p'b'}$ contains b (in which case it is vacuously true that $\overline{p'b'}$ intersects $\pi(z, b)$), as $\overline{p'b'}$ is tangent to $\pi(b, p)$ at p', if we move on $\overline{p'b'}$ from p' towards b', we will enter the interior of $\triangle(a, b, p)$ and let w be the first point on the boundary of $\triangle(a, b, p)$ we meet during the above movement after p' (i.e., we will go outside $\triangle(a, b, p)$ after w; e.g., see Fig. 14). Then, $\overline{p'w}$ separates z and b on its two sides, and thus $\overline{p'w}$ must intersect $\pi(z, b)$, say, at a point z'. We next prove $|\overline{z'b'}| \leq d(z', t)$. Indeed, $d(t, p') = |\overline{p'b'}| = |\overline{p'z'}| + |\overline{z'b'}|$. On the other hand, by triangle inequality, $d(t, p') \leq |\overline{p'z'}| + d(z', t)$. Hence, we obtain $|\overline{z'b'}| \leq d(z', t)$. Consequently, by triangle inequality, $|\overline{zb'}| \leq d(z, z') + |\overline{z'b'}| \leq d(z, z') + d(z', t) = d(z, t)$. This proves the claim. The above proves the first subcase $q \in \pi(a, p)$.

The second subcase $q \in \pi(a, p_{st})$. We next discuss the subcase $q \in \pi(a, p_{st})$; e.g., see Fig. 13 (b). The analysis is somewhat similar. First of all, we define b' in the same way as above; by the same argument, we have: (1) $\pi(p, p') \cup \overline{p'b'}$ is a convex chain and its length is equal to $d(p, b)$; (2) $|\overline{zb'}| \leq d(z, t)$.

The point a' is now defined in a slightly different way. Refer to Fig. 15. We extend $\overline{p'q}$ from p' to q until a point a' that satisfies the following two conditions: (1) $|\overline{qa'}| \geq d(s, q)$; (2) $\overline{p'a'}$ intersects $\pi(a, p)$, say, at a point q' (this is possible as $\overline{p'q}$ is tangent to $\pi(a, p_{st})$ at q). Note that $a' = q'$ if and only if $|\overline{qq'}| \geq d(s, q)$. Also note that $|\overline{qa'}| = d(s, q)$ if $a' \neq q'$. Next we prove the four properties of the geodesic triangle $\triangle(a', b', p)$.

First of all, notice that $|\overline{a'z}| = |\overline{a'q}| + |\overline{qz}| \geq d(s, q) + |\overline{qz}| = d(s, z)$. Recall that $|\overline{zb'}| \leq d(t, z) = d(s, z)$ and $d(s, z) \leq |\overline{zp'}|$. By a similar argument as before for the first subcase, the angle $\angle(a', b', p')$ is at least $\pi/2$. By the definitions of a' and q', $\pi(p, q') \cup \overline{q'a'}$ is a convex chain and we will show below that its length $l(a', p)$ is at most $d(s, p)$, which will prove all four properties of $\triangle(a', b', p)$.

To prove $l(a', p) \leq d(s, p)$, as $l(a', p) = |\overline{a'q'}| + d(q', p)$ and $d(s, p) = d(s, q') + d(q', p)$, it is sufficient to prove $|\overline{a'q'}| \leq d(s, q')$. If $a' = q'$, this is obviously true. We thus assume $a' \neq q'$. Then, $|\overline{qq'}| < |\overline{qa'}| = d(s, q)$. Hence, $d(s, p') =$

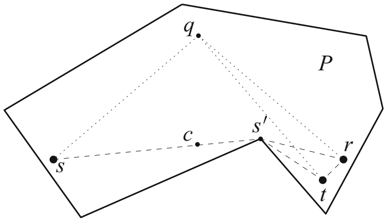

Fig. 16 Illustrating an example where α exists when $c \in \pi(s, s') \setminus \{s'\}$. The apexes of the geodesic triangle $\triangle(s, r, t)$ are s', $r' = r$, and $t' = t$. $|\overline{s't}| < |\overline{s'r}|$. c is the middle point of $\pi(s, r) = \overline{ss'} \cup \overline{s'r}$. However, one can verify (e.g., by a ruler) that q is equidistant to s, r, and t, and thus $\alpha = q$ exists

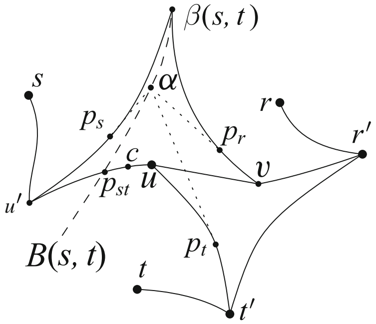

Fig. 17 Illustrating Lemma 4.10. In this example, $v' = v$

$d(s, q) + |\overline{qp'}| = |\overline{qa'}| + |\overline{qp'}| = |\overline{a'p'}| = |\overline{a'q'}| + |\overline{q'p'}|$. On the other hand, by triangle inequality, $d(s, p') \le d(s, q') + |\overline{q'p'}|$. Therefore, we obtain $|\overline{a'q'}| \le d(s, q')$. The above proves the second subcase $q \in \pi(a, p_{st})$. The lemma thus follows. □

4.2.2 The Case $c \in \pi(s, s') \setminus \{s'\}$

We now consider the case $c \in \pi(s, s') \setminus \{s'\}$. For this case, Oh and Ahn [22] (see the proof of [22, Lemma 3.6]) claimed that α does not exist. However, this is not correct; see Fig. 16 for a counterexample.

Let v be the vertex incident to s' in $\pi(s', r')$. To make the notation consistent with the previous subcase, we let $u = s'$. We have the following lemma (e.g., see Fig. 17), which is literally the same as Lemma 4.7 (the proof is also somewhat similar although there are some different arguments).

Lemma 4.10 (i) *α must be in the geodesic triangle $\triangle(s, r, \beta(s, t))$.*

(ii) *The apexes of $\triangle(s, r, \beta(s, t))$ are u', v', and $\beta(s, t)$, where u' (resp., v') is the junction vertex of and $\pi(s, r)$ and $\pi(s, \beta(s, t))$ (resp., $\pi(r, \beta(s, t))$) (in Fig. 17, $v' = v$).*

(iii) *p_s must be on the pseudo-convex chain $\pi(u', \beta(s, t)) \cup \pi(u', v')$ and $\overline{\alpha p_s}$ is tangent to the chain.*

(iv) *p_r must be on the pseudo-convex chain $\pi(v', \beta(s, t)) \cup \pi(v', u')$ and $\overline{\alpha p_r}$ is tangent to the chain.*

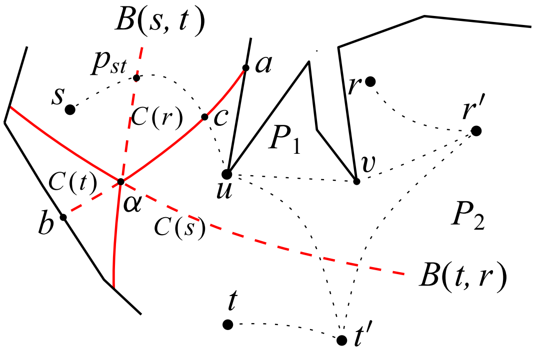

Fig. 18 Illustrating FVD(s, t, r), whose edges are depicted by thick (red) solid curves. The black solid segments are part of the boundary of P. The (red) dotted curves belong to bisectors but not on FVD(s, t, r). The red curve connecting a and b is $B(s, r)$; the portion between c and a (resp., b) is $B_1(s, r)$ (resp., $B_2(s, r)$). The point α is the only vertex of FVD(s, t, r) because it is equidistant from all three sites. The cells $C(s)$, $C(t)$, and $C(r)$ are ordered clockwise around α, while s, t, and r are ordered counterclockwise around the boundary of their geodesic hull, a contradiction

(v) *p_t must be on the pseudo-convex chain $\pi(t_{uv}, u) \cup \pi(t_{uv}, v)$ and $\overline{\alpha p_t}$ is tangent to the chain, where t_{uv} is the junction vertex of $\pi(t, u)$ and $\pi(t, v)$.*

(vi) *$\overline{\alpha p_t}$ intersects $\overline{uv}$.*

Proof As $c \in \pi(s, s') \setminus \{s'\}$ and $u = s'$, u cannot be s and thus must be a polygon vertex. But v can be either a polygon vertex or the site r. We assume that v is a polygon vertex since the other case can be reduced to this case by the same technique as in the proof of Lemma 4.8.

As both u and v are polygon vertices, $\overline{uv}$ divides P into two sub-polygons; one of them, denoted by P_1, does not contain t and we use P_2 to denote the other one. Let P' be the one of P_1 and P_2 that contains s. We will argue later that P' must be P_1.

We first show that the bisector $B(s, t)$ is in P'. Recall that p_{st} is the middle point of $B(s, t)$. Since $c \in \pi(s, s') \setminus \{s'\}$ and $d(s, c) = d(r, c) > d(c, t)$, $d(s, t) = d(s, c) + d(t, c) < d(s, c) + d(c, r) = d(s, r)$. Hence $d(s, p_{st}) = d(s, t)/2 < d(s, r)/2 = d(s, c)$. Thus, $p_{st} \in \pi(s, c) \setminus \{c\}$. Note that $\pi(s, c)$ is in P' since $\pi(s, c) \subseteq \pi(s, u)$ and $\pi(s, u) \in P'$ (the latter holds because both s and u are in P'). Therefore, $p_{st} \in P'$. Further, as $p_{st} \in \pi(s, c) \subseteq \pi(s, u) \setminus \{u\}$, $p_{st} \notin \overline{uv}$. Since $p_{st} \in \pi(s, r)$, $B(s, t)$ does not intersect $\pi(s, r)$ other than p_{st} by Observation 4.5. As $\overline{uv} \subseteq \pi(s, r)$, we obtain that $B(s, t)$ does not intersect $\overline{uv}$. Because p_{st} is in $B(s, t) \cap P'$ and $B(s, t)$ does not intersect $\overline{uv}$, $B(s, t)$ must be in P'.

Since $c \in \pi(s, s') \setminus \{s'\}$ and $c \in B(s, r)$, $B(s, r)$ is also in P' by following the analysis similar to the above. As α is equidistant from s, t, and r, α is on both $B(s, t)$ and $B(s, r)$. Therefore, α is in P'.

We next argue that P' must be P_1. We assume that s, t, r are ordered counterclockwise around the boundary of their geodesic hull (e.g., see Fig. 17). Assume to the contrary that P' is P_2 (e.g., see Fig. 18). The point c divides $B(s, r)$ into two portions, one going above $\pi(s, r)$ and the other going below $\pi(s, r)$ (we intuitively assume that $\pi(s, r)$ from s to r goes "horizontally" from left to right); let $B_1(s, r)$

(resp., $B_2(s,r)$) be the first (resp., second) portion (e.g., see Fig. 18, where the red curve between c and a is $B_1(s,r)$ and the red curve between c and b is $B_2(s,r)$). Since P' is P_2, the two polygon edges of P incident to u must be from the "above" of $\pi(s,r)$. Hence, for any point $p \in B_1(s,r)$, both shortest paths $\pi(r,p)$ and $\pi(t,p)$ must contain u. Thus, $d(r,p) = d(r,u) + d(u,p)$ and $d(t,p) = d(t,u) + d(u,p)$. Since $d(r,c) > d(t,c)$ and both $\pi(r,c)$ and $\pi(t,c)$ contain u, $d(r,u) > d(t,u)$ holds. Therefore, $d(r,p) > d(t,p)$. This implies that no point on $B_1(s,r)$ is equidistant from r and t, and thus $B_1(s,r)$ does not contain α. As $\alpha \in B(s,r)$, we have $\alpha \in B_2(s,r)$. Consider the farthest Voronoi diagram FVD(s,t,r) of the three sites s,t,r only (without considering other sites of S). Let $C(p)$ be the cell of $p \in \{s,t,r\}$ in the diagram. As $d(c,s) = d(c,r) > d(c,t)$, c belongs to the common boundary of $C(s)$ and $C(r)$, i.e., c is on an edge of FVD(s,t,r). The point α divides $B(s,r)$ into two portions, one of which contains c. The above implies that the portion of $B(s,r)$ containing c is an edge of FVD(s,t,r) (e.g., see Fig. 18). Recall that $p_{st} \in \pi(s,c) \setminus \{c\}$. Since $\pi(s,c) \subset \pi(s,r)$ and c is the middle point of $\pi(s,r)$, we obtain that $d(s,p_{st}) = d(t,p_{st}) < d(r,p_{st})$, implying that $p_{st} \in C(r)$. The point α partitions $B(s,t)$ into two portions, one of which contains p_{st}; since $p_{st} \in C(r)$, the portion of $B(s,t)$ containing p_{st} is not an edge of $F(s,t,r)$. Then, one can verify that the three cells $C(s)$, $C(t)$, and $C(r)$ in FVD(s,t,r) are ordered clockwise along the boundary of P (e.g., see Fig. 18). According to Aronov [3], s, t, and r should also be ordered clockwise around the boundary of their geodesic hull. But this contradicts the fact that s, t, and r are ordered counterclockwise around the boundary of their geodesic hull. The above proves that P' is P_1. Since $B(s,t) \in P'$ and $\alpha \in B(s,t)$, we obtain that $\alpha \in P_1$.

We next argue that α must be in the geodesic triangle $\triangle(s,r,\beta(s,t))$. The argument is similar to the proof of Lemma 4.7, so we briefly discuss it. To simplify the notation, let $q = \beta(s,t)$. Since $q \in B(s,t)$ and $B(s,t) \in P_1$, q is in P_1. By the same analysis as in the proof of Lemma 4.7, α is on $B(s,t)$ between p_{st} and q (e.g., see Fig. 17), and thus α is in the geodesic triangle $\triangle(s,t,q)$ and q is an apex of $\triangle(s,t,q)$. Since $t \in P_2$ and $q \in P_1$, $\pi(q,t)$ must cross $\overline{uv}$ at a point p, and thus α is also in $\triangle(s,p,q)$ and q is an apex of $\triangle(s,p,q)$. Further, since $p \in \overline{uv} \subseteq \pi(s,r)$, α is in $\triangle(s,r,q)$ and q is an apex of $\triangle(s,r,q)$. This proves the lemma statements (i) and (ii). The lemma statements (iii) and (iv) also immediately follow.

Finally, we argue that $\overline{\alpha p_t}$ intersects $\overline{uv}$. Assume to the contrary that this is not true. Since $\alpha \in P_1$ and $t \in P_2$, $\pi(\alpha,t)$ must cross $\overline{uv}$ at a point z. As $\overline{\alpha p_t}$ does not intersect $\overline{uv}$, p_t must be a polygon vertex in $\pi(\alpha,z)$, which is subpath of $\pi(\alpha,t)$. As $z \in \overline{uv} \subseteq \pi(s,r)$, $\pi(\alpha,z)$ must be "between" $\pi(\alpha,s)$ and $\pi(\alpha,r)$. Since no two paths of $\pi(\alpha,s)$, $\pi(\alpha,z)$, and $\pi(\alpha,r)$ cross each other and P is a simple polygon, p_t must be in either $\pi(\alpha,s)$ and $\pi(\alpha,r)$. As $p_t \in \pi(\alpha,t)$ and α is equidistant from s, t, and r, p_t is on the bisector between t and one of s and r. This contradicts our general position assumption since p_t is a vertex of P. The above proves that $\overline{\alpha p_t}$ intersects $\overline{uv}$, i.e., the lemma statement (vi), which also leads to the lemma statement (v). □

Due to the preceding lemma, our algorithm works as follows. First, we compute the vertices u', v', and t_{uv}, which can be done in $O(\log n)$ time by the GH data structure. Then we apply the tentative prune-and-search technique [17] on the three pseudo-convex chains specified in the lemma in a similar way as before to compute α in $O(\log n)$ time. Finally, we validate α in $O(\log n)$ time in a similar way as before. The overall time of the algorithm is $O(\log n)$. Lemma 3.1 is thus proved.

A summary of the algorithm for Lemma 3.1. Given three sites s, t, r as specified in Lemma 3.1, the goal is to compute $\alpha(s, t, r)$. We start with applying Lemma 4.3 to compute the geodesic center c of the three sites. We check whether c is equidistant to the three sites in $O(\log n)$ time. If yes, $\alpha(s, t, r) = c$ and we are done. Otherwise, we proceed to check whether one of the two cases in Lemma 4.6 happens. If yes, then $\alpha(s, t, r)$ is not a Voronoi vertex of FVD(S) and we simply return null. Otherwise, using the GH data structure, we determine in $O(\log n)$ time which of the following three cases holds: $c \in \pi(s', r')$, $c \in \pi(s, s') \setminus \{s'\}$, and $c \in \pi(r', r) \setminus \{r'\}$. In the first case $c \in \pi(s', r')$, we apply the algorithm for Lemma 4.7 to compute $\alpha(s, t, r)$. In the second case $c \in \pi(s, s') \setminus \{s'\}$, we apply the algorithm for Lemma 4.10 to compute $\alpha(s, t, r)$. The third case is symmetric to the second case, so the algorithm is similar. In any case, $\alpha(s, t, r)$ can be computed in $O(\log n)$ time.

Remark. As discussed above, there are two mistakes in the algorithm of Oh and Ahn [22, Lemma 3.6]: (1) In the subcase $c \in \pi(s', r')$, the case where not both u and v are polygon vertices is missed; (2) in the subcase $c \notin \pi(s', r')$, they erroneously claimed that α does not exist. Both mistakes can be corrected with our new results. Indeed, in both cases we have proved that $\overline{\alpha p_t}$ intersects $\overline{uv}$ (more specifically, it is proved in Lemma 4.7 for the first case and proved in Lemma 4.10 for the second case). With this critical property, their algorithm of [22, Lemma 3.6] (which was originally designed for the case where $c \in \pi(s', r')$ and both u and v are polygon vertices) can be applied to compute α in $O(\log^2 n)$ time. In this way, [22, Lemma 3.6] is remedied and thus all other results of [22] that rely on Lemma 3.6 are not affected.

5 Conclusions

In this paper, we presented a deterministic algorithm for computing the geodesic farthest-point Voronoi diagram for a set S of m points in a simple polygon P of n vertices. Our algorithm runs in $O(n + m \log m)$ time, matching the $\Omega(n + m \log m)$ lower bound. The space complexity of our algorithm is $O(n+m)$. Thus, our algorithm is optimal in both time and space.

As mentioned in Sect. 1, if all points of S are on the boundary of P, Barba [9] gave a randomized algorithm that solves the problem in $O(n + m)$ expected time. An interesting question is whether the algorithm can be derandomized, e.g., using some techniques and geometric observations derived in this paper. Also, for the problem in polygons with holes (i.e., P has holes), as discussed in Sect. 1, Bae and Chwa [6] gave an algorithm of $O(nm \log^2(n + m) \log m)$ time, and they also proved that $\Theta(nm)$ is the combinatorial complexity of the diagram in the worst case, implying that $\Omega(nm)$ is a lower bound for computing the diagram. Hence, there is still some gap between the upper and lower bounds. It would be interesting to see whether our techniques can be helpful to further improve the algorithm of [6].

Acknowledgements The author would like to thank three anonymous reviewers for their comments that helped improve the presentation of the paper.

Data Availibility Statement Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

References

1. Ahn, H.-K., Barba, L., Bose, P., De Carufel, J.-L., Korman, M., Oh, E.: A linear-time algorithm for the geodesic center of a simple polygon. Discrete Comput. Geom. **56**(4), 836–859 (2016)
2. Aronov, B.: On the geodesic Voronoĭ diagram of point sites in a simple polygon. Algorithmica **4**(1), 109–140 (1989)
3. Aronov, B., Fortune, S., Wilfong, G.: The furthest-site geodesic Voronoĭ diagram. Discrete Comput. Geom. **9**(3), 217–255 (1993)
4. Asano, T., Toussaint, G.: Computing the geodesic center of a simple polygon. In: Discrete Algorithms and Complexity (Kyoto 1986). Perspect. Comput., vol. 15, pp. 65–79. Academic Press, Boston (1987)
5. Avis, D.: On the complexity of finding the convex hull of a set of points. Discrete Appl. Math. **4**(2), 81–86 (1982)
6. Bae, S.W., Chwa, K.-Y.: The geodesic farthest-site Voronoi diagram in a polygonal domain with holes. In: 25th ACM Symposium on Computational Geometry (Aarhus 2009), pp. 198–207. ACM, New York (2009)
7. Bae, S.W., Korman, M., Okamoto, Y.: The geodesic diameter of polygonal domains. Discrete Comput. Geom. **50**(2), 306–329 (2013)
8. Bae, S.W., Korman, M., Okamoto, Y.: Computing the geodesic centers of a polygonal domain. Comput. Geom. **77**, 3–9 (2019)
9. Barba, L.: Optimal algorithm for geodesic farthest-point Voronoi diagrams. In: 35th International Symposium on Computational Geometry (Portland 2019). Leibniz International Proceedings in Informatics, vol. 129, # 12. Leibniz-Zentrum für Informatik, Wadern (2019)
10. Chazelle, B.: A theorem on polygon cutting with applications. In: 23rd Annual Symposium on Foundations of Computer Science (Chicago 1982), pp. 339–349. IEEE, New York (1982)
11. Chazelle, B., Edelsbrunner, H., Grigni, M., Guibas, L., Hershberger, J., Sharir, M., Snoeyink, J.: Ray shooting in polygons using geodesic triangulations. Algorithmica **12**(1), 54–68 (1994)
12. van Emde Boas, P.: On the $\Omega(n \log n)$ lower bound for convex hull and maximal vector determination. Inform. Process. Lett. **10**(3), 132–136 (1980)
13. Guibas, L.J., Hershberger, J.: Optimal shortest path queries in a simple polygon. J. Comput. System Sci. **39**(2), 126–152 (1989)
14. Hershberger, J.: A new data structure for shortest path queries in a simple polygon. Inform. Process. Lett. **38**(5), 231–235 (1991)
15. Hershberger, J., Suri, S.: Matrix searching with the shortest-path metric. SIAM J. Comput. **26**(6), 1612–1634 (1997)
16. Hershberger, J., Suri, S.: An optimal algorithm for Euclidean shortest paths in the plane. SIAM J. Comput. **28**(6), 2215–2256 (1999)
17. Kirkpatrick, D., Snoeyink, J.: Tentative prune-and-search for computing fixed-points with applications to geometric computation. Fund. Inform. **22**(4), 353–370 (1995)
18. Lee, D.T.: Farthest neighbor Voronoi diagrams and applications. Technical report # 80-11-FC-04. Northwestern University, Evanston (1980)
19. Liu, C.-H.: A nearly optimal algorithm for the geodesic Voronoi diagram of points in a simple polygon. Algorithmica **82**(4), 915–937 (2020)
20. Mitchell, J.S.B.: Geometric shortest paths and network optimization. In: Handbook of Computational Geometry, pp. 633–701. North-Holland, Amsterdam (2000)
21. Oh, E.: Optimal algorithm for geodesic nearest-point Voronoi diagrams in simple polygons. In: 30th Annual ACM-SIAM Symposium on Discrete Algorithms (San Diego 2019), pp. 391–409. SIAM, Philadelphia (2019)
22. Oh, E., Ahn, H.-K.: Voronoi diagrams for a moderate-sized point-set in a simple polygon. Discrete Comput. Geom. **63**(2), 418–454 (2020)

23. Oh, E., Barba, L., Ahn, H.-K.: The geodesic farthest-point Voronoi diagram in a simple polygon. Algorithmica **82**(5), 1434–1473 (2020)
24. Papadopoulou, E., Lee, D.T.: A new approach for the geodesic Voronoi diagram of points in a simple polygon and other restricted polygonal domains. Algorithmica **20**(4), 319–352 (1998)
25. Pollack, R., Sharir, M., Rote, G.: Computing the geodesic center of a simple polygon. Discrete Comput. Geom. **4**(1), 611–626 (1989)
26. Preparata, F.P., Shamos, M.I.: Computational Geometry. Texts and Monographs in Computer Science. Springer, New York (1985)
27. Suri, S.: Computing geodesic furthest neighbors in simple polygons. J. Comput. System Sci. **39**(2), 220–235 (1989)
28. Toussaint, G.: Computing geodesic properties inside a simple polygon. Rev. Intell. Artif. **3**(2), 9–42 (1989)
29. Wang, H.: On the geodesic centers of polygonal domains. J. Comput. Geom. **9**, 131–190 (2018)
30. Yao, A.C.C.: A lower bound to finding convex hulls. J. Assoc. Comput. Mach. **28**(4), 780–787 (1981)