Minimizing Distance-to-Sight in Polygonal Domains

Eunjin Oh
Max Planck Institute for Informatics Saarbrücken, Germany
eoh@mpi-inf.mpg.de

Abstract

In this paper, we consider the *quickest pair-visibility problem* in polygonal domains. Given two points in a polygonal domain with h holes of total complexity n, we want to minimize the maximum distance that the two points travel in order to see each other in the polygonal domain. We present an $O(n \log^2 n + h^2 \log^4 h)$-time algorithm for this problem. We show that this running time is almost optimal unless the 3SUM problem can be solved in $O(n^{2-\varepsilon})$ time for some $\varepsilon > 0$.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Visibility in polygonal domains, shortest path in polygonal domains

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2018.59

1 Introduction

Consider two mobile robots under the line-of-sight communication model [8, 15]. In this model, the two robots are required to be visible to each other in order to establish communication. In the case that they are not visible to each other, we can move one of them to see the other. Motivated by this model, the *quickest visibility problem* was introduced. In this problem, we are given a starting point s and a target point t amidst polygonal obstacles in the plane, and the objective is to find a shortest collision-free path for s to move along to see t. This problem can be solved in $O(n \log n)$ time, where n is the total complexity of the polygonal obstacles, by applying the *continuous Dijkstra paradigm* [10] as mentioned in [2].

Arkin et al. [2] studied the query variant of this problem. More precisely, given h polygonal obstacles (holes) of total complexity n and a target point, they presented a data structure of size $O(n^2 2^{\alpha(n)} \log n)$ so that the length of a shortest path for a query starting point to move along to see the target point can be computed in $O(K \log^2 n)$ time, where K is the size of the visibility polygon from the target point and $\alpha(n)$ is the inverse Ackermann function. Recently, it is improved by Wang [14]. His data structure has size of $O(n \log h + h^2)$ and supports $O(h \log h \log n)$ query time.

In this paper, we study the *quickest pair-visibility problem* in polygonal domains. In this problem, both starting and target points move to see each other. Precisely, given h polygonal obstacles of total complexity n and two points disjoint from the obstacles, we want to compute the minimum distance that the two points travel in order to see each other. Here, there are two variants of the problem, one for minimizing the maximum of the two travel distances and one for minimizing the sum of the two travel distances.

Wynters and Mitchell studied this problem [15] for both variants. For the min-max variant, they gave an $O(n^3 \log n)$-time algorithm using $O(n^2)$ space. For the min-sum variant, they gave an $O(nm)$-time algorithm using $O(m)$ space, where m is the number of edges in the visibility graph of the polygonal obstacles. Note that m is $\Theta(n^2)$ in the worst case. Very recently, Ahn et al. [1] considered a simpler version of the quickest pair-visibility problem in which two points are given in a simple polygon with no holes, and presented linear-time

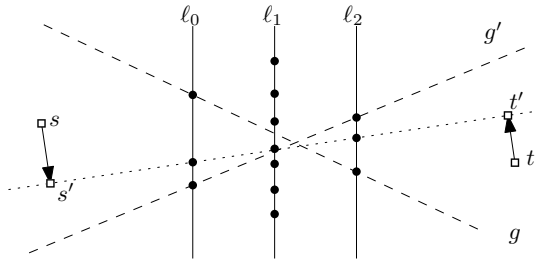


Figure 1 The quickest paths for s and t to see each other are ss' and tt'. Here, s' and t' are on a non-vertical line containing three input points.

algorithms for both the min-max and the min-sum variants of the problem. They also considered a query version of the problem for the min-max variant and presented a data structure supporting $O(\log^2 n)$ query time. Both the construction time and the space of the data structure are linear in the input size.

Our results. In this paper, we study the min-max variant of the quickest pair-visibility problem in a polygonal domain with h holes of total complexity n. We present an algorithm for this problem which takes $O(n \log^2 n + h^2 \log^4 h)$ time using $O(n \log n)$ space. This substantially improves the algorithm by Wynters and Mitchell, which takes $O(n^3 \log n)$ time using $O(n^2)$ space. Moreover, the running time of our algorithm is almost optimal unless the 3SUM problem can be solved in strongly subquadratic time. More specifically, the following lemma holds.

▶ **Lemma 1.** *Any algorithm for the min-max variant of the quickest pair-visibility problem in a polygonal domain with h holes of total complexity n takes $\Omega(n + h^{2-\varepsilon})$ time for any $\varepsilon > 0$ unless the* 3SUM *problem can be solved in $O(N^{2-\varepsilon})$, where N is the size of input for the* 3SUM *problem.*

Proof. We prove the lemma by introducing a reduction from a geometric version of the 3SUM problem. Given a set of n points with integer coordinates on three vertical lines $x = 0, x = 1$ and $x = 2$, the goal of the geometric version of the 3SUM problem is to determine whether there exists a non-vertical line containing three of the points. This problem is a 3SUM-hard problem in the sense that there is an $O(n)$-time reduction from the 3SUM problem to the geometric version of the 3SUM problem [7].

Given an instance of the geometric version of the 3SUM problem, we construct a polygonal domain as follows. Let S_i be the set of input points contained in the vertical line $\ell_i : x = i$ for $i = 0, 1, 2$. See Figure 1. Then $\ell_i \setminus S_i$ consists of $O(n)$ connected components (line segments or rays) in the plane. We consider each connected component as a hole of the polygonal domain. Let g be the line containing the topmost point of S_0 and the bottommost point of S_2. Similarly, let g' be the line containing the topmost point of S_2 and the bottommost point of S_0. We put s and t lying to the left of ℓ_0 and to the right of ℓ_2, respectively, so that $\max\{d_E(s, g), d_E(s, g'), d_E(t, g), d_E(t, g')\} \leq \min\{d_E(s, \ell_0), d_E(t, \ell_2)\}$, where $d_E(p, \ell)$ denotes the minimum Euclidean distance between a point p and a point in a line ℓ. Given g and g', such two points can be found in constant time.

Now consider the minimum of the maximum distance for s and t to travel in order to see each other. It is less than the minimum of $d(s, \ell_0)$ and $d(t, \ell_2)$ if and only if there is a non-vertical line containing three points of $S_0 \cup S_1 \cup S_2$. Therefore, if we can solve the quickest pair-visibility problem in $O(n + h^{2-\varepsilon})$ for some $\varepsilon > 0$, we can solve the geometric version of the 3SUM problem in $O(N^{2-\varepsilon})$ time, which proves the lemma. ◀

2 Preliminaries

Consider h disjoint simple polygons in the plane of total complexity n. Each polygon is considered as an open set. We let $\mathcal{P}$ be the set of the points in the plane not contained in any of the h polygons. Here we call $\mathcal{P}$ a *polygonal domain* and each polygon a *hole* of $\mathcal{P}$. We say that two points a and b in $\mathcal{P}$ are *visible* to each other if the line segment ab connecting a and b is contained in $\mathcal{P}$. For a set $A \subseteq \mathbb{R}^2$, we use ∂A and $\mathrm{int}(A)$ to denote the boundary and interior of A, respectively. We say a curve γ is *convex* if the Euclidean convex hull of γ contains γ on its boundary.

2.1 Geodesic Distance and Geodesic Disks

For two points a and b in $\mathcal{P}$, there might be more than one shortest path connecting a and b in $\mathcal{P}$. We use $d(a, b)$ to denote the length of a shortest path between a and b contained in $\mathcal{P}$, which we call the *geodesic distance* between a and b. The *shortest path map* of a point x, denoted by $\textsc{spm}(x)$, is the decomposition of $\mathcal{P}$ into cells such that for all points p within a cell the shortest paths between x and p have the same combinatorial structure. It has complexity of $O(n)$, and it can be constructed in $O(n \log n)$ time using $O(n \log n)$ space [10].

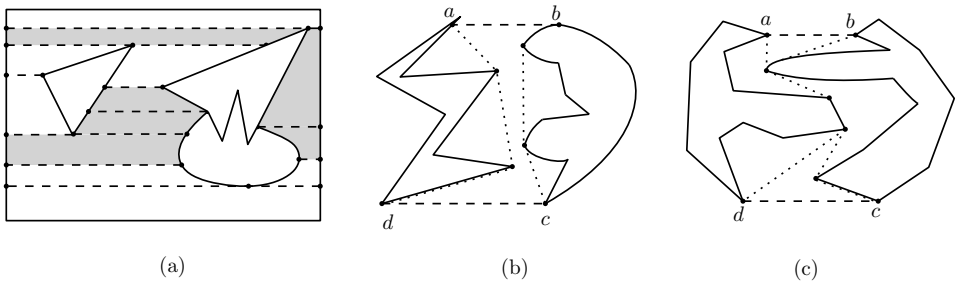
Given a point $x \in \mathcal{P}$ and a value $r \geq 0$, the *geodesic disk* of radius r centered at x, denoted by $\mathsf{D}_x(r)$, is defined as the set of all points in $\mathcal{P}$ whose geodesic distances from x are at most r. While $\mathsf{D}_x(r)$ is connected, its boundary is not necessarily connected. The boundary of $\mathsf{D}_x(r)$ consists of line segments and circular arcs. We call the endpoints of each maximal line segment and circular arc *vertices* of $\mathsf{D}_x(r)$. Consider the boundary of $\mathsf{D}_x(r)$ excluding the boundaries of all holes of $\mathcal{P}$ and the reflex vertices of $\mathsf{D}_x(r)$. Each connected component is the union of circular arcs of $\mathsf{D}_x(r)$. We call each connected component a *geodesic spiral* of $\mathsf{D}_x(r)$. Given $\textsc{spm}(x)$, we can construct $\mathsf{D}_x(r)$ for a fixed $r \geq 0$ in $O(n)$ time by considering all cells of $\textsc{spm}(x)$ one by one.

2.2 Extended Corridor Structure

Our algorithm uses the *extended corridor structure* of a domain [3, 4, 11]. A hole in the domain we will consider in this paper is either a simple polygon (a hole of $\mathcal{P}$) or a *splinegon* which is a part of $\mathsf{D}_s(r)$ and $\mathsf{D}_t(r)$ for two input points s and t in $\mathcal{P}$ and a fixed value $r \geq 0$. Each hole is considered as an open set. A splinegon is defined as a set obtained from a simple polygon P by replacing each edge e of P with a curved edge e' joining the endpoints of e such that the region bounded by e and e' is convex [6]. Thus a simple polygon itself is also a splinegon. A splinegon is said to be *simple* if an edge intersects another edge only at their common endpoint. A domain having splinegons as its holes is called a *splinegon domain*.

Chen and Wang [3] studied a decomposition of a splinegon domain $\mathcal{Q}$ which is called the *extended corridor structure*. They first considered a *bounded degree decomposition* of $\mathcal{Q}$, which is a subdivision of $\mathcal{Q}$ into cells each with at most four sides and with at most three neighboring cells. They presented an $O(n + h \log^{1+\varepsilon} h)$-time algorithm for computing a bounded degree decomposition of $\mathcal{Q}$ into $O(n)$ cells for any $\varepsilon > 0$, where h is the number of the splinegons in the domain and n is the total complexity of the splinegons. Such a subdivision is achieved by adding $O(n)$ nonintersecting diagonals. See Figure 2(a).

In our case, the boundary of a hole might overlap with the boundary of another hole while the holes are pairwise interior-disjoint. The algorithm by Chen and Wang still works for this case. In this case, an edge of the common boundary of two holes is considered as a (degenerate) cell. In this way, $\mathcal{Q}$ coincides with the union of the closures of the cells of the



(a) (b) (c)

Figure 2 (a) A bounded degree decomposition. The gray regions are junction regions with non-empty interiors. (b) An hourglass and four bays. (c) Two funnels, two canals and two bays.

bounded degree decomposition. The dual graph of the bounded degree decomposition is a planar graph such that the degree of each node is at most three. The cell corresponding to a node of degree 3 in the dual graph is called a *junction region*. It is known that the number of the junction regions in $\mathcal{Q}$ is $O(h)$ [3, 11].

Imagine that we remove the closures of all junction regions from $\mathcal{Q}$, which partitions $\mathcal{Q}$ into a number of connected regions. Each connected region is called a *corridor*. If a corridor has an empty interior, it lies on the common boundary of two holes. A corridor C with non-empty interior is a simple splinegon and has two boundary edges, say ab and cd, each incident to a junction region. We call them the *gates* of C. The boundary of C other than the gates consists of two chains connecting the gates such that each chain is a part of the boundary of a hole incident to C. See Figure 2(b–c).

Since a corridor is a simple splinegon, the shortest path connecting two points in C is unique. Let $\pi_C(x, y)$ denote the shortest path connecting two points x and y in C. For the gates ab and cd such that a, b, c, d appear on the boundary of C in the order, let H_C be the region bounded by ab, cd and $\pi_C(a, d)$ and $\pi_C(b, c)$. If $\pi_C(a, d)$ and $\pi_C(b, c)$ are disjoint, H_C is called an *hourglass* of C. See Figure 2(b). Otherwise, the interior of H_C consists of two connected components, each of which is called a *funnel* of C. See Figure 2(c). For both cases, we call a connected component R of $C \setminus H_C$ a *bay* if it is incident to exactly one edge of $\pi_C(a, d) \cup \pi_C(b, c)$. Otherwise, we call it a *canal*. We call an edge of $\pi_C(a, d) \cup \pi_C(b, c)$ incident to R (a bay or a canal) a *gate* of R. Also, we call $\pi_C(a, d)$ and $\pi_C(b, c)$ the *corridor paths*.

The union of the closures of the junction regions, hourglasses and funnels is called the *ocean* of $\mathcal{Q}$. It consists of $O(h)$ convex chains with a total of $O(n)$ vertices. Notice that the ocean is not necessarily connected. In this way, the interior of $\mathcal{Q}$ is subdivided into the ocean, bays and canals. We call this subdivision the *extended corridor structure* of $\mathcal{Q}$. Given a bounded degree decomposition of $\mathcal{Q}$, one can compute the extended corridor structure in $O(n)$ time [3]. This structure has been used as a tool for various types of visibility problems due to the following property.

▶ **Lemma 2** ([4, The Opaque Property]). *For any canal, suppose a line segment pq is in $\mathcal{Q}$ such that neither p nor q is in the canal. Then pq does not contain any point of the canal that is not on its two gates.*

Since the part of a corridor path incident to a canal is convex, we have the following lemma.

▶ **Lemma 3.** *For a canal, consider a line segment $pq \subset \mathcal{Q}$ intersecting a gate of the canal. Then pq intersects the part of a corridor path incident to the canal only at points on its gates.*

2.3 Sketch of the Algorithm

In this paper, we consider the min-max variant of the quickest pair-visibility problem. Given a polygonal domain $\mathcal{P}$ with h holes of complexity n and two points s and t in $\mathcal{P}$, the objective is to compute two paths in $\mathcal{P}$, one for s and one for t, to travel in order to see each other such that the maximum of the path lengths is minimized. Let r^* be the optimal solution, that is, the minimum of $\max\{d(s,s'), d(t,t')\}$ among all pairs (s',t') such that ss' and tt' are contained in $\mathcal{P}$. We first consider the decision problem that decides, given a real value $r \geq 0$, if there are two points s' and t' such that $d(s,s') \leq r$, $d(t,t') \leq r$, and s' and t' are visible to each other. Then to obtain r^*, we apply the parametric search technique by using the decision algorithm as a subprocedure.

For the decision problem, we assume that we have $\textsc{spm}(s)$ and $\textsc{spm}(t)$. Notice that they are independent of an input distance r. Also, we further assume that r is less than $d(s,t)/2$, that is, $\mathsf{D}_s(r)$ and $\mathsf{D}_t(r)$ are disjoint. Note that there is a point at distance $d(s,t)/2$ from each of s and t if $r \geq d(s,t)/2$. In this case, s and t can see each other by moving to this point. Therefore the answer is positive for any $r \geq d(s,t)/2$.

3 Decision Problem

For a fixed $r > 0$, the decision problem asks if there are two points s' and t' in $\mathcal{P}$ such that $d(s,s') \leq r$, $d(t,t') \leq r$, and s' and t' are visible to each other. If so, we say r is *feasible*. Such a segment $s't'$ always intersects geodesic spirals of $\mathsf{D}_s(r)$ and $\mathsf{D}_t(r)$. Thus there always exists a segment $s't' \subseteq \mathcal{P}$ intersecting $\mathsf{D}_s(r)$ only at s' and intersecting $\mathsf{D}_t(r)$ only at t' if r is feasible. We call such a segment for a feasible value r a *witness segment* for r. Notice that one endpoint of a witness segment lies on a geodesic spiral of D_s, and the other endpoint lies on a geodesic spiral of D_t. In this section, we present an $O(n + h^2 \log^2 h)$-time algorithm for deciding if a given value r is feasible. Since r is fixed, we simply let $\mathsf{D}_s = \mathsf{D}_s(r)$ and $\mathsf{D}_t = \mathsf{D}_t(r)$. Recall that D_s and D_t are disjoint by the assumption that $r < d(s,t)/2$.

3.1 Finding $O(h)$ Geodesic Spirals from Each Geodesic Disk

We want to construct the extended corridor structure of $\mathcal{P} \setminus (\mathrm{int}(\mathsf{D}_s) \cup \mathrm{int}(\mathsf{D}_t))$, but D_s and D_t are not necessarily splinegons because their boundaries are not necessarily connected. Moreover, it is possible that they have $\Theta(n)$ geodesic spirals even if $h = 1$. To decide if r is feasible efficiently, we choose a subset $\bar{\mathsf{D}}_s$ (and $\bar{\mathsf{D}}_t$) of D_s (and D_t) which is a splinegon containing $O(h)$ geodesic spirals of D_s (and D_t) on its boundary, and consider the extended corridor structure of $\mathcal{P} \setminus (\mathrm{int}(\bar{\mathsf{D}}_s) \cup \mathrm{int}(\bar{\mathsf{D}}_t))$.

Consider $\mathcal{P} \setminus \mathsf{D}_s$, which consists of $O(n)$ connected subregions. Since D_s and D_t are disjoint, D_t is contained in exactly one of such subregions. Moreover, no witness segment intersects subregions of $\mathcal{P} \setminus \mathsf{D}_s$ other than the one containing D_t. See Figure 3. Therefore, it suffices to consider the subregion containing D_t only. We can find the subregion containing D_t in $O(n)$ time as follows. Using $\textsc{spm}(s)$, we compute a shortest path connecting s and t, and find the point in the path whose distance from s is r. This point is contained in the boundary of the subregion. Starting from this point, we walk along the boundary of the subregion until we reach this point again in time linear in its complexity, which is $O(n)$.

Consider the common boundary of D_s and the subregion of $\mathcal{P} \setminus \mathsf{D}_s$ containing D_t. It is contained in a connected component, say η, of the boundary of D_s. Moreover, it consists of geodesic spirals of D_s appearing on η consecutively. Let a and b be the most clockwise and counterclockwise points on such geodesic spirals. Let $\bar{\mathsf{D}}_s$ denote the region bounded by

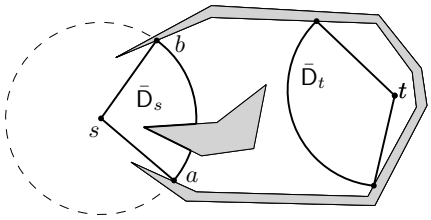


Figure 3 Every witness segment has its endpoints on $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$. Moreover, it does not intersect any other points of D_s and D_t.

a shortest path between s and a, a shortest path between s and b, and the part of η lying between a and b and containing the geodesic spirals on the common boundary. See Figure 3. We choose an arbitrary shortest path between s and a (or b) if it is not unique. Here, $\bar{\mathsf{D}}_s$ might contain a hole of $\mathcal{P}$. In this case, we ignore such holes. In the following, we assume that no hole of $\mathcal{P}$ intersects the interior of $\bar{\mathsf{D}}_s$. We define $\bar{\mathsf{D}}_t$ in the same way by changing the roles of s and t. Then a witness segment intersects D_s only at a point on $\bar{\mathsf{D}}_s$ and intersects D_t only at a point on $\bar{\mathsf{D}}_t$. Also, we have the following lemma.

▶ **Lemma 4.** *$\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$ contain $O(h)$ geodesic spirals of D_s and D_t, respectively, on their boundaries.*

Proof. We prove the lemma for $\bar{\mathsf{D}}_s$ only. The case of $\bar{\mathsf{D}}_t$ can be proved analogously.

An endpoint of a geodesic spiral of $\bar{\mathsf{D}}_s$ is a point on the boundary of a hole of $\mathcal{P}$ or a reflex vertex of D_s. We claim that for each hole H of $\mathcal{P}$, at most two geodesic spirals of $\bar{\mathsf{D}}_s$ have their endpoints on the boundary of H. We also claim that there are $O(h)$ reflex vertices of D_s lying on the boundary of $\bar{\mathsf{D}}_s$. These two claims imply the lemma.

For the first claim, consider a hole H of $\mathcal{P}$. Assume to the contrary that there are three geodesic spirals, say γ_1, γ_2 and γ_3, having their endpoints on the boundary of H. Recall that the geodesic spirals of $\bar{\mathsf{D}}_s$ are incident to the same connected component of $\mathcal{P} \setminus \mathsf{D}_s$, say R. Therefore, the boundary of H appears on the boundary of R at least twice. Notice that H is a simple polygon, which is connected. This means that D_s is disconnected, which is a contradiction.

For the second claim, let v be a reflex vertex of D_s lying on the boundary of $\bar{\mathsf{D}}_s$. Let β_1 and β_2 be the circular arcs of D_s incident to v. Consider a shortest path π_i connecting the center of β_i and s for $i = 1, 2$. If there are more than one shortest path, we choose the one so that the region bounded by π_1, π_2, and the two line segments connecting v and the centers of β_i is minimized. Such a region contains a hole of $\mathcal{P}$ by construction. Moreover, such regions for all reflex vertices of D_s lying on the boundary of $\bar{\mathsf{D}}_s$ are pairwise interior disjoint by the choice of π_i. Since each such region contains a hole of $\mathcal{P}$, there are at most h reflex vertices of D_s lying on the boundary of $\bar{\mathsf{D}}_s$. Therefore, the lemma holds. ◀

3.2 Extended Corridor Structure of the Splinegon Domain

We construct the extended corridor structure of $\mathcal{Q} = \mathcal{P} \setminus (\mathrm{int}(\bar{\mathsf{D}}_s) \cup \mathrm{int}(\bar{\mathsf{D}}_t))$ in $O(n + h \log^{1+\varepsilon} h)$ time for any $\varepsilon > 0$ [3]. Notice that we consider the *interiors* of $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$ as holes of $\mathcal{Q}$. The boundary of the ocean of $\mathcal{Q}$ consists of $O(h)$ convex curves each of which consists of a part of a single hole of $\mathcal{Q}$ or a part of a corridor path.

Recall that an (straight or circular) arc of the ocean is a part of the boundary of the holes of $\mathcal{Q}$ or a gate of a bay or a canal. Consider the arcs of the ocean which are gates of the bays and canals defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$. By construction, the boundary of each such

bay or canal consists of its gates and geodesic spirals of $\bar{\mathsf{D}}_s$ or $\bar{\mathsf{D}}_t$ only. Therefore, there are $O(h)$ such arcs of the ocean by Lemma 4 and the fact that a geodesic spiral contains a reflex vertex of D_s or D_t only at its endpoints. Imagine that we remove the gates of the bays and canals defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$ from the boundary of the ocean. The remaining part of the boundary still consists of $O(h)$ convex curves. Let Γ be the set of such convex curves and the gates of the bays and canals defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$. Note that the union of all curves and gates in Γ is the boundary of the ocean. A curve of Γ is *defined* by $\bar{\mathsf{D}}_s$ (or $\bar{\mathsf{D}}_t$) if it lies on the boundary of $\bar{\mathsf{D}}_s$ (or $\bar{\mathsf{D}}_t$) or it is a gate of a bay or a canal defined by $\bar{\mathsf{D}}_s$ (or $\bar{\mathsf{D}}_t$).

The following lemmas are keys of our decision algorithm. Due to them, it suffices to consider the curves of Γ only.

▶ **Lemma 5.** *If a witness segment does not intersect the closure of the ocean, it is contained in the interior of a corridor defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$. Moreover, if such a corridor exists, r is feasible.*

Proof. If a witness segment ℓ does not intersect the closure of the ocean, it is contained in a corridor, say C. By construction, the boundary of C consists of parts of the boundaries of two holes and two gates. Since the endpoints of ℓ are on geodesic spirals of $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$, the holes defining C are $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$. Now assume that a corridor defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$ exists. Then a gate of the corridor connects a point of a geodesic spiral of $\bar{\mathsf{D}}_s$ and a point of a geodesic spiral of $\bar{\mathsf{D}}_t$. In other words, such a gate is a witness segment, and thus r is feasible. ◀

▶ **Lemma 6.** *If a witness segment ℓ intersects the closure of the ocean, the intersection between ℓ and the ocean is a line segment whose endpoints are on curves of Γ defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$.*

Proof. Let ℓ' be the intersection between ℓ and the ocean. By construction, a connected component of $\ell \setminus \ell'$ is contained in a bay or a canal by the opaque property. Thus $\ell \setminus \ell'$ consists of at most two connected components, and ℓ' is a line segment. Let p be an endpoint of ℓ'. If p is an endpoint of ℓ, it lies on a convex curve of Γ contained on a geodesic spiral of $\bar{\mathsf{D}}_s$ or $\bar{\mathsf{D}}_t$, and the lemma holds. Thus we assume that p is not an endpoint of ℓ. Then the connected component of $\ell \setminus \ell'$ incident to p is contained in a bay or a canal defined by $\bar{\mathsf{D}}_s$ or $\bar{\mathsf{D}}_t$. This means that p lies on a gate of the bay or canal, and therefore, it lies on the curve of Γ defined by $\bar{\mathsf{D}}_s$ or $\bar{\mathsf{D}}_t$. ◀

▶ **Lemma 7.** *If r is feasible and no corridor is defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$, there is a witness segment ℓ such that the intersection between ℓ and the ocean is tangent to a curve of Γ or connects an endpoint of a curve of Γ defined by $\bar{\mathsf{D}}_s$ and an endpoint of a curve of Γ defined by $\bar{\mathsf{D}}_t$.*

Proof. Let ℓ' be the intersection between ℓ and the ocean. Its endpoints are contained in curves γ_s and γ_t of Γ defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$, respectively, by Lemma 6. We move one endpoint of ℓ' in clockwise direction along the curve γ_s of Γ containing it until (1) ℓ' (excluding its endpoints) contains a vertex of the ocean, (2) ℓ' intersects γ_s (or γ_t) at a point other than the endpoints of ℓ', or (3) the endpoint reaches an endpoint of γ_s. For Case (1), ℓ' is tangent to a curve of Γ, and thus we are done. For Case (2), ℓ' is tangent to γ_s (or γ_t), and thus we are done. For Case (3), we move the other endpoint of ℓ' in the same way. Then the lemma holds. ◀

We call the intersection between a witness segment satisfying Lemma 7 and the ocean an *ocean-restricted witness segment*.

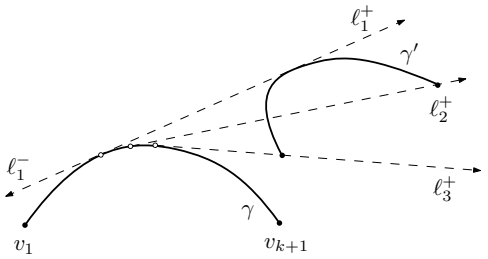


Figure 4 Any ray tangent to γ lying between ℓ_1^+ and ℓ_2^+ intersects γ' twice and any ray tangent to γ lying between γ_2^+ and γ_3^+ intersects γ' once.

3.3 Finding a Witness Segment: Rotating Lines around Convex Curves

We assume that no corridor is defined by $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$. Otherwise, we return the positive answer immediately by Lemma 5. Our goal in this subsection is to find an ocean-restricted witness segment if r is feasible. By definition (and by Lemma 7), an ocean-restricted witness segment is tangent to a convex curve of Γ or contains an endpoint of a curve of Γ. We check for each convex curve γ of Γ if there is an ocean-restricted witness segment tangent to γ. In a similar way, we check for each endpoint of the curves of Γ if it contains an ocean-restricted witness segment.

Imagine that we rotate a line tangent to γ along γ. More specifically, let $\langle e_1, \ldots, e_k \rangle$ be the sequence of the (circular or straight) arcs of γ in order. For an integer i, let v_i and v_{i+1} be the endpoints of e_i. The process will be initialized with the line tangent to e_1 at v_1. It rotates along e_1 until it hits v_2. Then it rotates around v_2 (while remaining tangent to γ at v_2) until it is tangent to e_2. In general, the current line is rotated around v_i in a way so that it remains tangent to γ at v_i until it is tangent to e_i, and then it rotates along e_i until it hits v_{i+1}. The process is iterated with v_{i+1} as the new rotation center. The process terminates as soon as the line is tangent to γ at v_{k+1}. If an ocean-restricted witness segment is tangent to γ, we encounter the line containing it during the process.

In the following, for each line ℓ we encounter during the process, we let ℓ^+ and ℓ^- be the connected components (rays) of $\ell \setminus \gamma$ such that ℓ^+ goes towards v_{k+1} and ℓ^- goes towards v_1. See Figure 4. An ocean-restricted witness segment is contained in ℓ if and only if the first curve of Γ hit by ℓ^+ is defined by one of D_s and D_t, and the first curve of Γ hit by ℓ^- is defined by the other geodesic disk. We show how to maintain the first curve of Γ hit by ℓ^+ only. We can do this for ℓ^- analogously.

More generally, we maintain the sequence S of the curves of Γ hit by ℓ^+ in order. Since every curve of Γ is convex, it appears on S at most twice. During the sweep, for each convex curve γ' of Γ, there are at most four events where the number of appearances of γ' on S changes. See Figure 4. Moreover, such events (rays) are on common tangents of γ and γ' or lines tangent to γ which pass through an endpoint of γ'. We can compute the common tangents of γ and γ' $O(\log h)$ time if the arcs of each convex curve are stored in a balanced binary search tree [13]. Similarly, we can compute the lines tangent to γ and passing through a specific point in $O(\log h)$ time [13]. We can construct the balanced binary search trees of the curves of Γ in time linear in their complexities after computing Γ. Since no convex curve of Γ crosses another convex curve of Γ, the sequence S changes only at these events. Thus there are $O(h)$ events in total, and we can obtain and sort all events in $O(h \log h)$ time. We can handle each event in $O(\log^2 n)$ time as follows.

When we encounter a new event ℓ^+, the convex curve γ' defining ℓ^+ disappears from S or appears on S. For the case that it disappears from S, we update S accordingly in $O(\log n)$ time. For the other case, we apply binary search on the elements of S to find the position of the new appearance of γ on S. In each iteration of the binary search, we want to compute the order of the points (at most four points) in $\ell^+ \cap \gamma'$ and $\ell^+ \cap \gamma''$ along ℓ^+ for some curve γ'' appearing on the current sequence S. We can compute the points in $O(\log n)$ time by a straightforward binary search on the arcs of γ' (and γ''), and then sort them in $O(1)$ time. This gives the position of the new appearance of γ' with respect to γ''. After $O(\log n)$ iterations, we can find the position of the new appearance of γ' on S. Then we update S accordingly. In this way, we can handle each event in $O(\log^2 n)$ time.

After rotating a line along γ, we can check if an ocean-restricted witness segment is contained in a line we encountered so far. Since we have $O(h)$ curves of Γ, the total time for checking if an ocean-restricted witness segment is tangent to a curve of Γ is $O(h^2 \log^2 h)$.

Also, we check for each endpoint of the curves of Γ if it contains an ocean-restricted witness segment. We can do this in a similar way: rotate a line around this endpoint. Therefore, we have the following lemma.

▶ **Lemma 8.** *Given a value $r > 0$, we can check if there are two points s' and t' such that $d(s, s') \leq r$, $d(t, t') \leq r$, and s' and t' are visible to each other in $O(n + h^2 \log^2 h)$ time assuming that $\mathrm{SPM}(s)$ and $\mathrm{SPM}(t)$ are given.*

4 Optimization Problem

Let (s^*, t^*) be a pair of points in $\mathcal{P}$ that minimizes the maximum of $d(s, s^*)$ and $d(t, t^*)$ such that s^* and t^* are visible to each other. Let r^* be the maximum of $d(s, s^*)$ and $d(t, t^*)$. In this section, we compute (s^*, t^*) and r^* by applying parametric search technique [12].

Basically, we apply the decision algorithm with input r^* without explicitly computing r^*. In the decision algorithm, we maintain a number of structures including geodesic disks, the splinegon domain $\mathcal{Q}$ and the sequence Γ which depend on an input distance r. In this section, we consider such structures as functions of r. For example, we use $\Gamma(r)$ to denote the sequence Γ for an input distance r. While the algorithm described in this section is executed, we maintain an interval $[r_1, r_2)$ containing r^* so that the *combinatorial structures* of structures we have computed so far remain the same for every $r \in [r_1, r_2)$. Then we will see that r_1 becomes r^* for the interval we have at the end.

4.1 Combinatorial Structures of $\bar{\mathsf{D}}_s$ and $\bar{\mathsf{D}}_t$

The first step of the decision algorithm is to compute $\mathsf{D}_s(r)$, $\mathsf{D}_t(r)$, $\bar{\mathsf{D}}_s(r)$ and $\bar{\mathsf{D}}_t(r)$. Here, we compute their *combinatorial structures* for $r = r^*$ instead of computing them explicitly.

We first compute the combinatorial structures of $\mathsf{D}_s(r^*)$ and $\mathsf{D}_t(r^*)$. Notice that the endpoints of the circular arcs of $\mathsf{D}_s(r^*)$ and $\mathsf{D}_t(r^*)$ lie on edges of $\mathrm{SPM}(s)$ and $\mathrm{SPM}(t)$, respectively. Also, the boundary of $\mathsf{D}_s(r^*)$ is not necessarily connected. For a radius $r > 0$, the combinatorial structure of $\mathsf{D}_s(r)$ is defined as a set of the sequences of edges of $\mathrm{SPM}(s)$ such that each sequence consists of the edges of $\mathrm{SPM}(s)$ intersecting a connected component of the boundary of $\mathsf{D}_s(r)$ in the clockwise order along the component.

For each vertex of $\mathrm{SPM}(s)$, we compute the geodesic distance between the vertex and s. Also for each edge of $\mathcal{P}$, we compute the smallest geodesic distance between a point on the edge and s. We can compute them in $O(n)$ time by considering all cells of $\mathrm{SPM}(s)$ one by one. Then we sort all distances in increasing order in $O(n \log n)$ time. We find

the smallest interval $[r_1, r_2)$ containing r^* for two distances r_1 and r_2 we obtained by applying binary search on all such distances with the decision algorithm. Since the decision algorithm takes $O(n + h^2 \log^2 h)$ time assuming that we have $\text{SPM}(s)$ and $\text{SPM}(t)$, we can find $[r_1, r_2)$ in $O(n \log n + h^2 \log^2 h \log n) = O(n \log n + h^2 \log^3 h)$ time in total. For any radius $r \in [r_1, r_2)$, the combinatorial structure of $\mathsf{D}_s(r)$ remains the same. We also compute the combinatorial structure of $\mathsf{D}_t(r)$, and update $[r_1, r_2)$ containing r^* so that for any $r \in [r_1, r_2)$, the combinatorial structure of $\mathsf{D}_t(r)$ (and $\mathsf{D}_s(r)$) remains the same.

Also, we define the combinatorial structure of $\bar{\mathsf{D}}_s(r)$ to be the sequence of the edges of $\text{SPM}(s)$ intersecting the boundary of $\bar{\mathsf{D}}_s(r)$ in clockwise order. For any $r \in [r_1, r_2)$, the combinatorial structure of $\bar{\mathsf{D}}_s(r)$ remains the same by the definition of $\bar{\mathsf{D}}_s(r)$. The same holds for $\bar{\mathsf{D}}_t(r)$.

By construction, an endpoint of a circular arc of $\bar{\mathsf{D}}_s(r)$ is represented as an algebraic function of constant complexity for a value $r \in [r_1, r_2)$. We obtain the splinegon domain $\mathcal{Q}(r)$ defined by $\text{int}(\bar{\mathsf{D}}_s(r))$, $\text{int}(\bar{\mathsf{D}}_t(r))$ and the holes of $\mathcal{P}$ for $r \in [r_1, r_2)$. Here, a vertex of $\mathcal{Q}$ is represented as an algebraic function with respect to r.

4.2 Combinatorial Structure of the Extended Corridor Structure

To construct the extended corridor structure of a splinegon domain, the algorithm by Chen and Wang [3] computes a bounded degree decomposition of the splinegon domain. Then based on this, they compute the extended corridor structure. In the following, we split each arc of $\mathcal{Q}(r)$ into at most four pieces so that it is monotone with respect to the x-axis and y-axis. In other words, we add at most three vertices to each arc. Here each of the new vertices is also represented as an algebraic function with respect to r.

Bounded degree decomposition. The algorithm by Chen and Wang [3] first decomposes the domain with respect to the horizontal extensions obtained from each hole vertex of $\mathcal{P}$ going in both directions until they hit the boundary of the domain. Then each cell has at most four sides, but it might have more than three neighboring cells. In such a case, the algorithm splits each such cell further with respect to vertical extensions from vertices of the cell. Then it splits each of the resulting cells further with respect to its diagonal if it still has more than three neighboring cells. In our case, we want to represent the vertices of the bounded degree decomposition of $\mathcal{Q}(r)$ as algebraic functions with respect to r for $r \in [r'_1, r'_2)$ for some interval $[r'_1, r'_2) \subseteq [r_1, r_2)$ containing r^*.

Suppose that the order of the vertices of $\mathcal{Q}(r)$ with respect to the y-axis is the same for any $r \in [r'_1, r'_2)$ for some interval $[r'_1, r'_2) \subseteq [r_1, r_2)$ containing r^*. Moreover, the order of the vertices of $\mathcal{Q}(r)$ with respect to the x-axis is the same for any $r \in [r'_1, r'_2)$. Then the combinatorial structure of the bounded degree decomposition of $\mathcal{Q}(r)$ remains the same for any $r \in [r'_1, r'_2)$. In other words, a vertex of the bounded degree decomposition of $\mathcal{Q}(r)$ is represented as an algebraic function of r for $r \in [r'_1, r'_2)$. Thus in the following, we show how to sort the vertices of $\mathcal{Q}(r)$ with respect to the x-axis.

To do this, we use Cole's sorting algorithm which sorts m elements in $O(\log m)$ iterations each consisting of $O(m)$ comparisons [5]. Here, the comparisons in each iteration is independent to one another. In our case, we are to sort all vertices of $\mathcal{Q}(r)$ with respect to the x-axis. Here, $m = O(n)$. For each iteration, we complete $O(n)$ comparisons of vertices of $\mathcal{Q}(r)$ as follows. Suppose that we are to compare two vertices $v_1(r)$ and $v_2(r)$ represented by algebraic functions of $r \in [r_1, r_2)$. The result of the comparison changes only at $O(1)$ times as r changes from r_1 to r_2. We obtain $O(1)$ such values in $O(1)$ time. We do this for all of the $O(n)$ comparisons, and then we have $O(n)$ values. Then we apply binary

search on the values so that we find an interval $[r'_1, r'_2) \subseteq [r_1, r_2)$ containing r^* and the result of each comparison remains the same for any $r \in [r'_1, r'_2)$. We can find the interval in $O(T_p \log n) = O(n \log n + h^2 \log^3 h)$ time, where T_p is the running time for the decision algorithm. We update the interval $[r_1, r_2)$ that we maintain to $[r'_1, r'_2)$. After completing $O(\log n)$ iterations, we obtain the interval $[r_1, r_2)$ containing r^* such that the sorted list of the vertices of $\mathcal{Q}(r)$ remains the same for every $r \in [r_1, r_2)$. Thus we can sort all vertices with respect to the x-axis (and the y-axis) in $O(n \log^2 n + h^2 \log^4 h)$ time in total, and we can obtain the combinatorial structure of the bounded degree decomposition of $\mathcal{Q}(r^*)$ in the same time.

Extended corridor structure. The next step for computing the extended corridor structure is to compute the shortest paths for each corridor. We can make this procedure parallelized using the algorithm [9]. More precisely, this algorithm computes the shortest path between two points in $O(\log N)$ iterations each consisting of $O(N)$ comparisons. As we did for the bounded degree decomposition, we can reduce the interval $[r_1, r_2)$ containing r^* so that the combinatorial structure of the extended corridor structure remains the same for any $r \in [r_1, r_2)$. Since this can be done in $O(\log n)$ iterations each consisting of $O(n)$ steps for all corridors of $\mathcal{Q}(r^*)$, we can compute the combinatorial structure of the extended corridor structure in $O(n \log^2 n + h^2 \log^4 h)$ time as we did for computing the bounded degree decomposition.

▶ **Lemma 9.** *We can obtain the combinatorial structure of the extended corridor structure of $\mathcal{Q}(r^*)$ in $O(n \log^2 n + h^2 \log^4 h)$ time.*

Since we have the combinatorial structure of $\mathcal{Q}(r^*)$, we can obtain $\Gamma(r^*)$ in the same time. Again, an endpoint of each convex curve of $\Gamma(r)$ is represented as an algebraic function of r.

4.3 Finding a Witness Segment

The last step of the decision algorithm is to rotate a line along each curve of $\Gamma(r^*)$. We have $O(h^2)$ events in total each of which is either a common tangent between two curves of $\Gamma(r^*)$ or the line tangent to a curve of $\Gamma(r^*)$ and passing through an endpoint of another curve of $\Gamma(r^*)$. A common tangent between two curves of $\Gamma(r)$ is defined by a pair of arcs from two curves. More precisely, it is a common tangent of two arcs of the two curves or a line passing through endpoints of the two curves. Instead of computing the common tangents, we compute the pairs defining them. Since we can compute a common tangent between two convex curves in $O(\log n)$ time and we have $O(h^2)$ pairs of convex curves, we can compute all events in $O(\log n)$ iterations each consisting of $O(h^2)$ steps. As we did before, we can complete each iteration in $O(T_p \log(h^2) + h^2) = O(n \log n + h^2 \log^3 h)$ time, where T_p denotes the running time of the decision algorithm. Therefore, we can obtain $[r_1, r_2)$ such that the set of the pairs of arcs defining the events remains the same for every $r \in [r_1, r_2)$ in $O(n \log^2 n + h^2 \log^4 h)$ time. Similarly, we can do this for the events of the other type.

This means that for any $r \in [r_1, r_2)$, the first curve of $\Gamma(r)$ hit by ℓ^+ remains the same for any line ℓ tangent to a curve of $\Gamma(r)$. Therefore, the answer of the decision problem remains the same for any $r \in [r_1, r_2)$. Since r^* is contained in $[r_1, r_2)$, the answer is positive for every $r \in [r_1, r_2)$. By definition, we have $r^* = r_1$. Thus we have the following theorem.

▶ **Theorem 10.** *Given a polygonal domain $\mathcal{P}$ with h holes of total complexity n, we can compute two points s' and t' minimizing $\max\{d(s, s'), d(t, t')\}$ such that s' and t' are visible to each other in $O(n \log^2 n + h^2 \log^4 h)$ time.*

References

1. Hee-Kap Ahn, Eunjin Oh, Lena Schlipf, Fabian Stehn, and Darren Strash. On Romeo and Juliet Problems: Minimizing Distance-to-Sight. In *Proceedings of the 16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2018)*, 2018.
2. Esther M. Arkin, Alon Efrat, Christian Knauer, Joseph S. B. Mitchell, Valentin Polishchuk, Günter Rote, Lena Schlipf, and Topi Talvitie. Shortest Path to a Segment and Quickest Visibility Queries. *Journal of Computational Geometry*, 7(2):77–100, 2016.
3. Danny Z. Chen and Haitao Wang. Computing Shortest Paths Among Curved Obstacles in the Plane. *ACM Transactions on Algorithms*, 11(4):26:1–26:46, 2015.
4. Danny Z. Chen and Haitao Wang. Computing the Visibility Polygon of an Island in a Polygonal Domain. *Algorithmica*, 77(1):40–64, 2017.
5. Richard Cole. Parallel Merge Sort. *SIAM Journal on Computing*, 17(4):770–785, 1988.
6. David P. Dobkin and Diane L. Souvaine. Computational geometry in a curved world. *Algorithmica*, 5(1):421–457, 1990.
7. Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Compututational Geometry: Theory and Applications*, 45(4):140–152, 2012.
8. Anurag Ganguli, Jorge Cortes, and Francesco Bullo. Visibility-based multi-agent deployment in orthogonal environments. In *Proceedings of American Control Conference*, pages 3426–3431, 2007.
9. Michael T. Goodrich, Steven B. Shauck, and Sumanta Guha. Parallel methods for visibility and shortest-path problems in simple polygons. *Algorithmica*, 8(1):461–486, 1992.
10. John Hershberger and Subhash Suri. An optimal algorithm for Euclidean shortest paths in the plane. *SIAM Journal on Computing*, 28(6):2215–2256, 1999.
11. S. Kapoor, S. N. Maheshwari, and J. S. B. Mitchell. An Efficient Algorithm for Euclidean Shortest Paths Among Polygonal Obstacles in the Plane. *Discrete & Computational Geometry*, 18(4):377–383, 1997.
12. Nimrod Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM*, 30(4):852–865, 1983.
13. Mark H. Overmars and Jan van Leeuwen. Maintenance of configurations in the plane. *Journal of Computer and System Sciences*, 23(2):166–204, 1981.
14. Haitao Wang. Quickest Visibility Queries in Polygonal Domains. In *Proceedings of the 33rd International Symposium on Computational Geometry (SoCG 2017)*, volume 77, pages 61:1–61:16, 2017.
15. Erik L. Wynters and Joseph S. B. Mitchell. Shortest Paths for a Two-Robot Rendez-Vous. In *Proceedings of the 5th Canadian Conference on Computational Geometry (CCCG 1993)*, pages 216–221, 1993.