An Optimal Algorithm to Compute the Inverse Beacon Attraction Region

Irina Kostitsyna
Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands
i.kostitsyna@tue.nl

Bahram Kouhestani
School of Computing, Queen's University, Kingston, Ontario, Canada
kouhesta@cs.queensu.ca

Stefan Langerman[1]
Département d'informatique, Université Libre de Bruxelles, Brussels, Belgium
stefan.langerman@ulb.ac.be

David Rappaport
School of Computing, Queen's University, Kingston, Ontario, Canada
daver@cs.queensu.ca

Abstract

The *beacon model* is a recent paradigm for guiding the trajectory of messages or small robotic agents in complex environments. A *beacon* is a fixed point with an attraction pull that can move points within a given polygon. Points move greedily towards a beacon: if unobstructed, they move along a straight line to the beacon, and otherwise they slide on the edges of the polygon. The Euclidean distance from a moving point to a beacon is monotonically decreasing. A given beacon *attracts* a point if the point eventually reaches the beacon.

The problem of attracting all points within a polygon with a set of beacons can be viewed as a variation of the art gallery problem. Unlike most variations, the beacon attraction has the intriguing property of being asymmetric, leading to separate definitions of *attraction region* and *inverse attraction region*. The attraction region of a beacon is the set of points that it attracts. It is connected and can be computed in linear time for simple polygons. By contrast, it is known that the inverse attraction region of a point – the set of beacon positions that attract it – could have $\Omega(n)$ disjoint connected components.

In this paper, we prove that, in spite of this, the total complexity of the inverse attraction region of a point in a simple polygon is linear, and present a $O(n \log n)$ time algorithm to construct it. This improves upon the best previous algorithm which required $O(n^3)$ time and $O(n^2)$ space. Furthermore we prove a matching $\Omega(n \log n)$ lower bound for this task in the algebraic computation tree model of computation, even if the polygon is monotone.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases beacon attraction, inverse attraction region, algorithm, optimal

Digital Object Identifier 10.4230/LIPIcs.SoCG.2018.55

Related Version A full version of this paper is available at [12], `http://arxiv.org/abs/1803.05946`.

[1] Directeur de Recherches du F.R.S.-FNRS.

34th International Symposium on Computational Geometry (SoCG 2018).
Editors: Bettina Speckmann and Csaba D. Tóth; Article No. 55; pp. 55:1–55:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

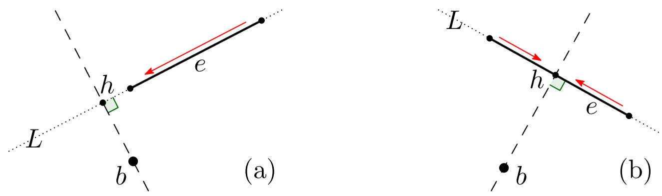

Figure 1 Points on an edge e slide towards the orthogonal projection h of the beacon on the supporting line of e.

1 Introduction

Consider a dense network of sensors. In practice, it is common that routing between two nodes in the network is performed by greedy geographical routing, where a node sends the message to its closest neighbor (by Euclidean distance) to the destination [11]. Depending on the geometry of the network, greedy routing may not be successful between all pairs of nodes. Thus, it is essential to determine nodes of the network for which this type of routing works. In particular, given a node in the network, it is important to compute all nodes that can successfully send a message to (or receive a message from) the input node. Motivated by this application Biro *et al.* [3] introduced the beacon routing model.

Let P be a simple polygon with n vertices. A *beacon* b is a point in P that can induce an attraction pull towards itself within P. The attraction of b causes points in P to move towards b as long as their Euclidean distance is maximally decreasing. As a result, a point p moves along the ray $\overrightarrow{pb}$ until it either reaches b or an edge of P. In the latter case, p slides on the edge towards h, the orthogonal projection of b on the supporting line of the edge (Figure 1). Note that among all points on the supporting line of the edge, h has the minimum Euclidean distance to b.

We say b *attracts* p, if p eventually reaches b. Interestingly, beacon attraction is not symmetric. The *attraction region* of b, denoted by $AR(b)$, is the set of all points in P that b attracts[2]. The *inverse attraction region* of a point p, denoted by $IAR(p)$, is the set of all beacon positions in P that can attract p.

The study of beacon attraction problems in a geometric domain, initiated by Biro *et al.* [3], finds its root in sensor networks, where the limited capabilities of sensors makes it crucial to design simple mechanisms for guiding their motion and communication. For instance, the beacon model can be used to represent the trajectory of small robotic agents in a polygonal domain, or that of messages in a dense sensor network. Using greedy routing, the trajectory of a robot (or a message) from a sender to a receiver closely follows the attraction trajectory of a point (the sender) towards a beacon (the receiver). However, greedy routing may not be successful between all pairs of nodes. Thus, it is essential to characterize for which pairs of nodes of the network for which this type of routing works. In particular, given a single node, it is important to compute the set of nodes that it can successfully receive messages from (its attraction region), and the set of node that it can successfully send messages to (its inverse attraction region).

In 2013, Biro *et al.* [5] showed that the attraction region $AR(b)$ of a beacon b in a simple polygon P is simple and connected, and presented a linear time algorithm to compute $AR(b)$.

[2] We consider the attraction region to be closed, *i.e.*, b attracts all points on the boundary of $AR(b)$.

Computing the inverse attraction region has proven to be more challenging. It is known [5] that the inverse attraction region $IAR(p)$ of a point p is not necessarily connected and can have $\Theta(n)$ connected components. Kouhestani *et al.* [14] presented an algorithm to compute $IAR(p)$ in $O(n^3)$ time and $O(n^2)$ space. In the special cases of monotone and terrain polygons, they showed improved algorithms with running times $O(n \log n)$ and $O(n)$ respectively.

In this paper, we prove that, in spite of not being connected, the inverse attraction region $IAR(p)$ always has total complexity[3] $O(n)$. Using this fact, we present the first optimal $O(n \log n)$ time algorithm for computing $IAR(p)$ for any simple polygon P, improving upon the previous best known $O(n^3)$ time algorithm. Since this task is at the heart of other algorithms for solving beacon routing problems, this improves the time complexity of several previously known algorithms such as approximating minimum beacon paths and computing the weak attraction region of a region [5].

To prove the optimality of our algorithm, we show an $\Omega(n \log n)$ lower bound in the algebraic computation tree model and in the bounded degree algebraic decision tree model, even in the case when the polygon is monotone.

Due to space limitations some of the proofs are omitted and can be found in the full version of this paper [12].

Related work

Greedy routing has been studied extensively in the literature of sensor network as a local (and therefore inexpensive) protocol for message sending. As a result, many applications in wireless and sensor networks utilize greedy routing to choose the next hop in their message sending protocol [10]. In the geometric domain, greedy routing has been studied in both discrete and continuous spaces. Bose *et al.* [6] studied routing problems in ad hoc wireless networks modeled as unit graphs and Kermarrec and Tan [11] presented an approximation algorithm to decompose a polygon into minimum number of routable regions, *i.e.*, regions in which greedy routing always works. Beacon routing, discussed in this paper, is essentially greedy routing in a polygonal environment representing an infinitely dense sensor network.

Several geometric problems related to the beacon model have been studied in recent years. Biro *et al.* [3] studied the minimum number of beacons necessary to successfully route between any pair of points in a simple n-gon P. This can be viewed as a variant of the art gallery problem, where one wants to find the minimum number of beacons whose attraction regions cover P. They proved that $\lceil \frac{n}{2} \rceil$ beacons are sometimes necessary and always sufficient, and showed that finding a minimum cardinality set of beacons to cover a simple polygon is NP-hard. For polygons with holes, Biro *et al.* [4] showed that $\lceil \frac{n}{2} \rceil - h - 1$ beacons are sometimes necessary and $\lceil \frac{n}{2} \rceil + h - 1$ beacons are always sufficient to guard a polygon with h holes. Combinatorial results on the use of beacons in orthogonal polygons have been studied by Bae *et al.* [1] and by Shermer [17]. Biro *et al.* [5] presented a polynomial time algorithm for routing between two fixed points using a discrete set of candidate beacons in a simple polygon and gave a 2-approximation algorithm where the beacons are placed with no restrictions. Kouhestani *et al.* [15] give an $\mathrm{O}(n \log n)$ time algorithm for beacon routing in a 1.5D polygonal terrain.

Kouhestani *et al.* [13] showed that the length of a successful beacon trajectory is less than $\sqrt{2}$ times the length of a shortest (geodesic) path. In contrast, if the polygon has internal holes then the length of a successful beacon trajectory may be unbounded.

[3] Total number of vertices and edges of all connected components.

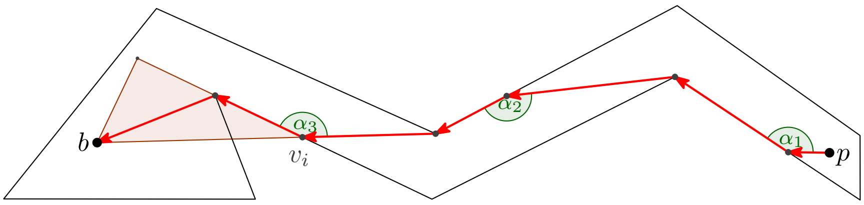

Figure 2 The angle between a straight movement towards the beacon and the following slide movement is always greater than $\pi/2$.

2 Preliminaries

A *dead point* $d \neq b$ is defined as a point that remains stationary in the attraction pull of b. The set of all points in P that eventually reach (and stay) on d is called the *dead region* of b with respect to d. A *split edge* is defined as the boundary between two dead regions, or a dead region and $AR(b)$. In the latter case, we call the split edge a *separation edge*.

If beacon b attracts a point p, we use the term *attraction trajectory*, denoted by $AT(p, b)$, to indicate the movement path of a point p from its original location to b. The attraction trajectory alternates between a straight movement towards the beacon (a *pull edge*) and a sequence of consecutive sliding movements (*slide edges*), see Figure 2.

▶ **Lemma 1.** *Consider the attraction trajectory $AT(p, b)$ of a point p attracted by beacon b. Let α_i denote the angle between the i-th pull edge and the next slide edge on $AT(p, b)$. Then α_i is greater than $\pi/2$.*

Note that, similarly, the angle between the i-th pull edge and the previous slide edge is also greater than $\pi/2$.

Let r be a reflex vertex of P with adjacent edges e_1 and e_2. Let H_1 be the half-plane orthogonal to e_1 at r, that contains e_1. Let H_2 be the half-plane orthogonal to e_2 at r, that contains e_2. The *deadwedge* of r (deadwedge(r)) is defined as $H_1 \cap H_2$ (Figure 3). Let b be a beacon in the deadwedge of r. Let ρ be the ray from r in the direction $\overrightarrow{br}$ and let s be the line segment between r and the first intersection of ρ with the boundary of P. Note that in the attraction of b, points on different sides of s have different destinations. Thus, s is a split edge for b. We say r *introduces* the split edge s for b to show this occurrence. Kouhestani *et al.* [14] proved the following lemma.

▶ **Lemma 2** (Kouhestani *et al.* [14])**.** *A reflex vertex r introduces a split edge for the beacon b if and only if b is inside the deadwedge of r.*

Let p and q be two points in a polygon P. We use $\overline{pq}$ to denote the straight-line segment between these points. Denote the shortest path between p and q in P (the geodesic path)

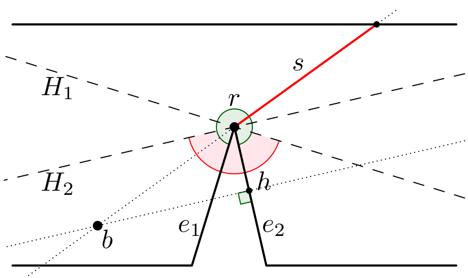

Figure 3 The deadwedge of r is shown by the red angle.

as $SP(p, q)$. The union of shortest paths from p to all vertices of P is called the *shortest path tree* of p, and can be computed in linear time [9] when P is a simple polygon. In our problem, we are only interested in shortest paths from p to reflex vertices of P. Therefore, we delete all convex vertices and their adjacent edges in the shortest path tree of p to obtain the *pruned shortest path tree* of p, denoted by $SPT_r(p)$.

A *shortest path map* for a given point p, denoted as $SPM(p)$, is a subdivision of P into regions such that shortest paths from p to all the points inside the same region pass through the same set of vertices of P [16]. Typically, shortest path maps are considered in the context of polygons with holes, where the subdivision represents grouping of the shortest paths of the same topology, and the regions may have curved boundaries. In the case of a simple polygon, the boundaries of $SPM(p)$ are straight-line segments and consist solely of the edges of P and extensions of the edges of $SPT_r(p)$. If a triangulation of P is given, it can be computed in linear time [9].

▶ **Lemma 3.** *During the movement of p on its beacon trajectory, the shortest path distance of p away from its original location monotonically increases.*

3 The structure of inverse attraction regions

The $O(n^3)$ time algorithm of Kouhestani *et al.* [14] to compute the inverse attraction region of a point p in a simple polygon P constructs a line arrangement A with quadratic complexity that partitions P into regions, such that, either all or none of the points in a region attract p. Arrangement A, contains three types of lines:

1. Supporting lines of the deadwedge for each reflex vertex of P,
2. Supporting lines of edges of $SPT_r(p)$,
3. Supporting lines of edges of P.

▶ **Lemma 4** (Kouhestani *et al.* [14])**.** *The boundary edges of $IAR(p)$ lie on the lines of arrangement A.*

Let $\overline{uv}$ be an edge of $SPT_r(p)$, where $u = parent(v)$. We associate three lines of the arrangement A to $\overline{uv}$: supporting line of $\overline{uv}$ and the two supporting lines of the deadwedge of v. By focusing on the edge $\overline{uv}$, we study the local effect of the reflex vertex v on $IAR(p)$, and we show that:

1. Exactly one of the associated lines to $\overline{uv}$ may contribute to the boundary of $IAR(p)$. We call this line the *effective associated line* of $\overline{uv}$ (Figure 4).
2. The effect of v on the inverse attraction region can be represented by at most two half-planes, which we call the *constraining half-planes* of $\overline{uv}$. These half-planes are bounded by the effective associated line of $\overline{uv}$.
3. Each constraining half-plane has a *domain*, which is a subpolygon of P that it affects. The points of the constraining half-plane that are inside the domain subpolygon cannot attract p (see the next section).

Our algorithm to compute the inverse attraction region uses $SPM(p)$. For each region of $SPM(p)$, we compute the set of constraining half-planes with their domain subpolygons containing the region. Then, we discard points of the region that cannot attract p by locating points which belong to at least one of these constraining half-planes.

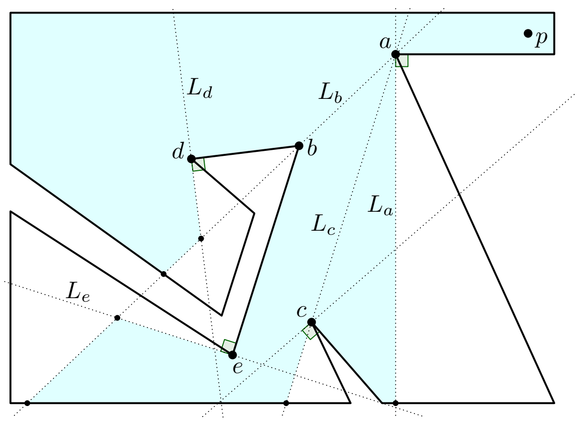

Figure 4 An example of an inverse attraction region with effective associated lines to each reflex vertex. Points in the colored region attract p. Here L_a, L_b, L_c, L_d and L_e are respectively the associated lines of the reflex vertices a, b, c, d and e.

Constraining half-planes

Let $\overline{uv}$ be an edge of $SPT_r(p)$, where $u = parent(v)$. We extend $\overline{uv}$ from u until we reach w, the first intersection with the boundary of P. Segment $\overline{uw}$ partitions P into two subpolygons. Let P_p be the subpolygon that contains p. Any path from p to any point in $P \setminus P_p$ passes through $\overline{uw}$. Thus a beacon outside of P_p that attracts p, must be able to attract at least one point on the line segment $\overline{uw}$. In order to determine the local attraction behaviour caused by the vertex v, and to find the effective line associated to $\overline{uv}$, we focus on the attraction pull on the points of $\overline{uw}$ (particularly the vertex u) rather than p. By doing so we detect points that cannot attract u, or any point on $\overline{uw}$, and mark them as points that cannot attract p. In other words, for each edge $\overline{uv} \in SPT_r(p)$ we detect a set of points in P that cannot attract u locally due to v. The attraction of these beacons either causes u to move to a wrong subpolygon, or their attraction cannot move u past v (see the following two cases for details). Later in Theorem 8, we show that this suffices to detect all points that cannot attract p.

Let e_1 and e_2 be the edges incident to v. Let H_1 be the half-plane, defined by a line orthogonal to e_1 passing through v, which contains e_1, and let H_2 be the half-plane, defined by a line orthogonal to e_2 passing through v, which contains e_2. Depending on whether u is in $H_1 \cup H_2$, we consider two cases:

Case 1. Vertex u is not in $H_1 \cup H_2$ (Figure 5). We show that in this case the supporting line of $\overline{uv}$ is the only line associated to v that may contribute to the boundary of $IAR(p)$, *i.e.*, it is the effective line associated to $\overline{uv}$. Let q be an arbitrary point on the open edge e_1. As u is not in $H_1 \cup H_2$, the angle between the line segments $\overline{uq}$ and $\overline{qv}$ is less than $\pi/2$. Consider an arbitrary attraction trajectory that moves u straight towards q. By Lemma 1, any slide movement of this attraction trajectory on the edge e_1 moves away from v. Now consider q to be on the edge e_2. Similarly any slide on the edge e_2 moves away from v. Thus, the line segment $\overline{uv}$ can only be crossed once in an attraction trajectory of u (and, similarly, of any other point on the line segment $\overline{uw}$). Note that this crossing movement happens via a pull edge. We use this observation to detect a set of points that do not attract u and thus do not attract p.

Now consider the supporting line L of the edge $\overline{uv}$. As u is not in $H_1 \cup H_2$, L partitions

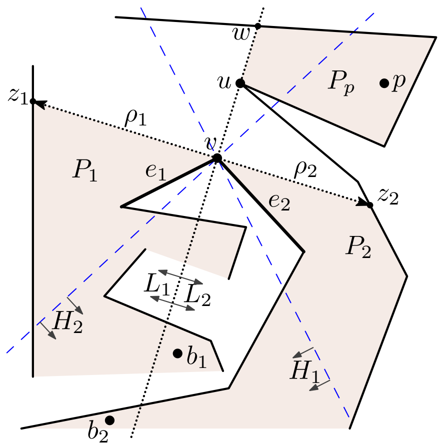

Figure 5 Vertex $u \notin H_1 \cup H_2$. Subpolygon P_2 is the domain of the constraining half-plane H_1, and P_1 is the domain of the constraining half-plane H_2.

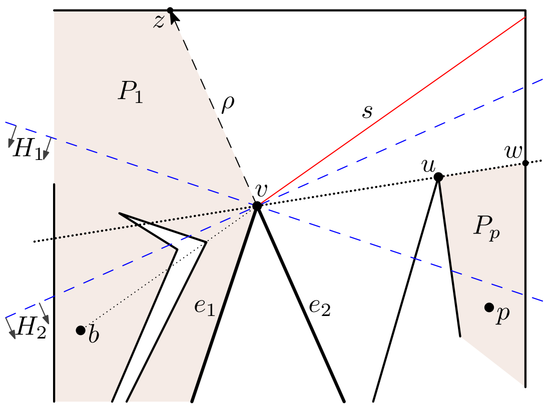

Figure 6 Vertex $u \in H_1 \cup H_2$. Subpolygon P_1 is the domain of the constraining half-plane H_2.

the plane into two half-planes L_1 containing the edge e_1, and L_2 containing the edge e_2. Without loss of generality, assume that the parent of u in $SPT_r(p)$ lies inside L_2 (refer to Figure 5). Recall that $\overline{uw}$ partitions P into two subpolygons, and P_p is the subpolygon containing p. We define subpolygons P_1 and P_2 as follows. Let ρ_1 be the ray originating at v, perpendicular to L in L_1, and let z_1 be the first intersection point of ρ_1 with the boundary of P. Define P_1 as the subpolygon of P induced by $\overline{vz_1}$ that contains the edge e_1. Similarly, let ρ_2 be the ray originating at v, perpendicular to L inside L_2, and let z_2 be the first intersection point of ρ_2 with the boundary of P. Define P_2 as the subpolygon of P induced by $\overline{vz_2}$ that contains the edge e_2. We provide the details of the following two lemmas in the full version of this paper [12].

▶ **Lemma 5.** *No point in $P_1 \cap L_2$ can attract p.*

▶ **Lemma 6.** *No point in $P_2 \cap L_1$ can attract p.*

In summary, in case 1, the effect of $\overline{uv}$ is expressed by two half-planes: L_2, affecting the subpolygon P_1, and L_1, affecting the subpolygon P_2. We call L_1 and L_2 the *constraining half-planes* of $\overline{uv}$, and we call P_1 and P_2 the *domain* of the constraining half-planes L_2 and L_1, respectively. Furthermore, we call $P_1 \cap L_2$ and $P_2 \cap L_1$ the *constraining regions* of $\overline{uv}$. Later we show that L is the only effective line associated to $\overline{uv}$.

Case 2. Vertex u is in $H_1 \cup H_2$ (refer to Figure 6). Without loss of generality assume u can see part of the edge e_2. Similar to the previous case, we define the subpolygon P_p; let w be the first intersection of the ray $\overrightarrow{vu}$ with the boundary of P. Note that $\overline{uw}$ partitions P into two subpolygons. Let P_p be the subpolygon containing p. Now let ρ be the ray originating at v, along the extension of edge e_2. Let z be the first intersection of ρ with the boundary of P. We use P_1 to denote the subpolygon induced by $\overline{vz}$ that contains e_1. We detect points in P_1 that cannot move u (past v) into P_1.

▶ **Lemma 7.** *No point in $P_1 \cap H_2$ can attract p.*

In summary, in case 2, the effect of $\overline{uv}$ on $IAR(p)$ can be expressed by the half-plane H_2. We call H_2 the *constraining half-plane* of $\overline{uv}$, P_1 the *domain* of H_2 and we call $P_1 \cap H_2$ the *constraining region* of $\overline{uv}$. Later we show that the supporting line of H_2 is the only effective line associated to v.

By combining these two cases, we prove the following theorem.

▶ **Theorem 8.** *A beacon b can attract a point p if and only if b is not in a constraining region of any edge of $SPT_r(p)$.*

Proof. By Lemmas 5, 6 and 7, if b is in the constraining region of an edge $\overline{uv} \in SPT_r(p)$ then it does not attract p.

Now let b be a point that cannot attract p. We will show that b is in the constraining region of at least one edge of $SPT_r(p)$. Let s be the separation edge of $AR(b)$ such that b and p are in different subpolygons induced by s (see, for example, Figure 6). Note that as the attraction region of a beacon is connected [2], there is exactly one such separation edge. Let v be the reflex vertex that introduces s and let u be the parent of v in $SPT_r(p)$. By Lemma 2, b is in the deadwedge of v. In addition, as the attraction region of a beacon is connected, b attracts v. We claim that b is in a constraining region of the edge $\overline{uv} \in SPT_r(p)$. First, we show that b cannot attract u. Consider $SP(p, u)$, the shortest path from p to u. If $SP(p, u)$ crosses s at some point q then u cannot be the parent of v in $SPT_r(p)$, because we can reach v with a shorter path by following $SP(p, u)$ from p to q and then reaching v from q. Therefore, $SP(p, u)$ does not cross s, so p and u are in the same subpolygon of P induced by s. As b does not attract p, we conclude that b does not attract u.

Consider the two cases: u is in $H_1 \cup H_2$ or not. We show that in each case, b is in a constraining region of $\overline{uv}$.

Case 1. Vertex u is not in $H_1 \cup H_2$ (refer to Figure 5). Let L be the supporting line of $\overline{uv}$, and similar to the previous case analysis let L_1 and L_2 be the constraining half-planes, and let P_1 and P_2 be the domains of L_2 and L_1, respectively. Without loss of generality, assume that b is in the half-plane L_2. We show that then b belongs to P_1.

As $b \in L_2$, the separation edge s extends from v into L_1, *i.e.*, $s \in L_1$. Then the point p and subpolygon P_2 lie on one side of s, and subpolygon P_1 lies on the other side of s. As beacon b does not attract p, the point p and the beacon b lie on different sides of s, and thus the beacon b and subpolygon P_1 lie on the same side of s.

We will show now that indeed $b \in P_1$. Beacon b attracts v and is in the deadwedge of v. Thus, in the attraction of b, v will enter P_1 via a slide move. We claim that v cannot leave P_1 afterwards. Consider the supporting line of ρ_1 which is a line orthogonal to $\overline{uv}$ at v. As u is not in $H_1 \cup H_2$, and the deadwedge of v is equal to $H_1 \cap H_2$, the deadwedge of v completely lies to one side of the supporting line. Therefore, in the attraction of v by any beacon inside the deadwedge of v, any point $q \neq v$ on $\overline{vz_1}$ moves straight towards the beacon along the ray $\overrightarrow{qb}$. In other words, in the attraction pull of b no point inside P_1 can leave P_1. Therefore, $b \in P_1$ and thus $b \in P_1 \cap L_2$. By definition, b belongs to a constraining region of $\overline{uv}$.

Case 2. Vertex u is in $H_1 \cup H_2$ (refer to Figure 6). Without loss of generality let $u \in H_2$. Consider the separation edge s. As the beacon b does not attract u, they lie on the opposite sides of s. As b is in the deadwedge of v, it is also in H_2, the constraining half-plane of $\overline{uv}$. Similar to the previous case, as b attracts v, $AT(v, b)$ never crosses ρ to leave P_1 and therefore, b is in P_1. Thus, $b \in P_1 \cap H_2$ and it belongs to the constraining region of $\overline{uv}$. ◀

▶ **Corollary 9.** *Consider the edge $\overline{uv} \in SPT_r(p)$. If u is not in $H_1 \cup H_2$ (case 1), then among three associated lines to $\overline{uv}$ only the supporting line of $\overline{uv}$ may contribute to the boundary of $IAR(p)$. If u is in $H_1 \cup H_2$ (case 2), then among three associated lines to $\overline{uv}$ only the*

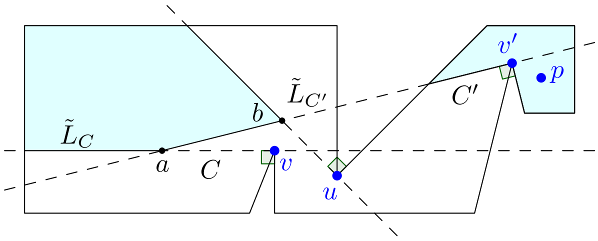

Figure 7 The charging scheme: vertex a is charged to the constraining half-plane C of vertex v. The inverse attraction region of p is the shaded region.

supporting line of H_2 may contribute to the boundary of $IAR(p)$, where H_2 is the half-plane orthogonal to the incident edge of v that u can partially see.

4 The complexity of the inverse attraction region

In this section we show that in a simple polygon P the complexity of $IAR(p)$ is linear with respect to the size of P.

We classify the vertices of the inverse attraction region into two groups: 1) vertices that are on the boundary of P, and 2) internal vertices. We claim that there are at most a linear number of vertices in each group. Throughout this section, without loss of generality, we assume that no two constraining half-planes of different edges of the shortest path tree are co-linear. Note that we can reach such a configuration with a small perturbation of the input points, which may just add to the number of vertices of $IAR(p)$.

Biro [2] showed that the inverse attraction region of a point in a simple polygon P is convex with respect to[4] P. Therefore, we have at most two vertices of $IAR(p)$ on each edge of P, and thus there are at most a linear number of vertices in the first group.

We use the following property of the attraction trajectory to count the number of vertices in group 2.

▶ **Lemma 10.** *Let L be the effective line associated to the edge $\overline{uv} \in SPT_r(p)$, where $u = parent(v)$. Let b be a beacon on $L \cap deadwedge(v)$ that attracts p. Then the attraction trajectory of p passes through both u and v.*

Next we define an ordering on the constraining half-planes. Let C be a constraining half-plane of the edge $\overline{uv} \in SPT_r(p)$ ($u = parent(v)$), and let C' be a constraining half-plane of the edge $\overline{u'v'} \in SPT_r(p)$ ($u' = parent(v')$). We say $C \leq C'$ if and only if $|SP(p, v)| \leq |SP(p, v')|$ (refer to Figure 7).

We use a charging scheme to count the number of internal vertices. An internal vertex resulting from the intersection of two constraining half-planes C and C' is charged to C' if $C \leq C'$, otherwise it is charged to C. In the remaining of this section, we show that each constraining half-plane is charged at most twice. Let P_C and P'_C denote the constraining regions related to C and C', respectively. And let L_C and $L_{C'}$ denote the supporting lines of C and C', respectively. In the previous section we showed that the line segments $L_C \cap P_C$ are the only parts of L_C that may contribute to the boundary of $IAR(p)$. Let $s \in L_C \cap P_C$ be a segment outside of the deadwedge of v. The next lemma shows that s does not appear on the boundary of $IAR(p)$, and we can ignore s when counting the internal vertices of $IAR(p)$.

[4] A subpolygon $Q \subseteq P$ is *convex with respect to* the polygon P if the line segment connecting two arbitrary points of Q either completely lies in Q or intersects P.

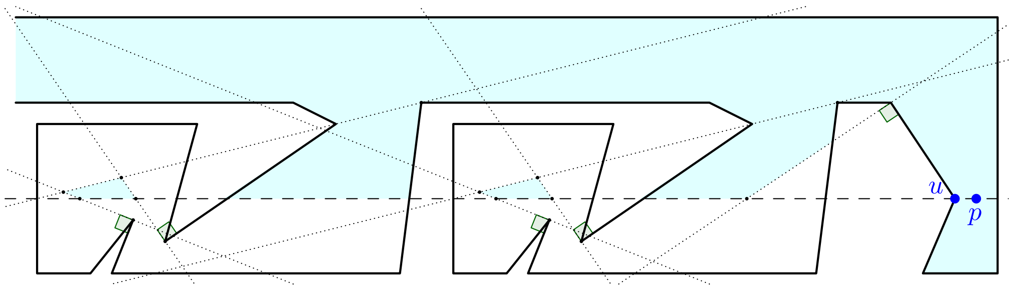

Figure 8 A constraining half-plane may contribute $O(n)$ vertices of group 2 to the inverse attraction region. Here the inverse attraction region of p is colored.

▶ **Lemma 11.** *Let $s \in L_C \cap P_C$ be a segment outside of the deadwedge of v. Then s (or a part of s with a non-zero length) does not appear on the boundary of $IAR(p)$.*

We define $\tilde{L}_C = L_C \cap P_C \cap \text{deadwedge}(v)$ and $\tilde{L}_{C'} = L_{C'} \cap P_{C'} \cap \text{deadwedge}(v')$. By Lemma 11, $\tilde{L}_C$ and $\tilde{L}_{C'}$ are the subset of L_C and $L_{C'}$ that may appear on the boundary of $IAR(p)$, therefore, the intersection points of all $\tilde{L}_C$ and $\tilde{L}_{C'}$ are the only possible locations for internal vertices of $IAR(p)$. Consider an internal vertex a resulting from the intersection of $\tilde{L}_C$ and $\tilde{L}_{C'}$.

▶ **Lemma 12.** *Let $a = \tilde{L}_C \cap \tilde{L}_{C'}$ be an internal vertex of $IAR(p)$ and let $C' \leq C$ (Figure 7). Then all points on $\tilde{L}_C$ are in the domain of C'.*

We charge a to C if $C' \leq C$, otherwise we charge it to C'. Assume a is charged to C. By Lemma 12, all points on $\tilde{L}_C$ to one side of a belong to the domain of C' and therefore are in C'. Thus, C cannot contribute any other internal vertices to this side of a. This implies that C can be charged at most twice (once from each end) and as there are a linear number of constraining half-planes, we have at most a linear number of vertices of group 2, and we have the following theorem.

▶ **Theorem 13.** *The inverse attraction region of a point p has linear complexity in a simple polygon.*

Note that, as illustrated in Figure 8, a constraining half-plane may contribute many vertices of group 2 to the inverse attraction region, but nevertheless it is charged at most twice.

5 Computing the inverse attraction region

In this section we show how to compute the inverse attraction region of a point inside a simple polygon in $O(n \log n)$ time.

Let region R_i of the shortest path map $SPM(p)$ consist of all points t such that the last segment of the shortest path from p to t is $\overline{v_i t}$ (Figure 9). Vertex v_i is called the *base* of R_i. Extend the edge of $SPT_r(p)$ ending at v_i until the first intersection z_i with the boundary of P. Call the segment $w_i = \overline{v_i z_i}$ a *window*, and point z_i – the *end* of the window; window w_i is a boundary segment of R_i.

We will construct a part of the inverse attraction region of p inside each region of the shortest path map $SPM(p)$ independently. A point in a region of $SPM(p)$ attracts p only if its attraction can move p into the region through the corresponding window.

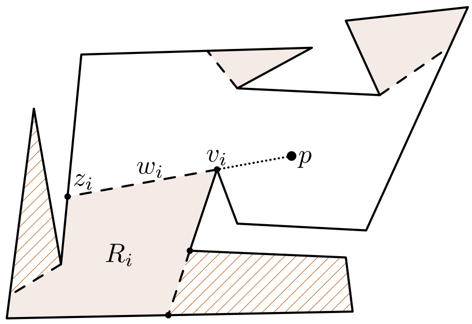

Figure 9 R_i is a region of $SPM(p)$ with base v_i. Segment w_i is the window, and z_i – its end.

▶ **Lemma 14.** *Let R_i be a region of $SPM(p)$ with a base vertex v_i. If v_i lies in some domain subpolygon P_e, then any point t in R_i lies in P_e.*

Let R_i be a region of $SPM(p)$ with a base vertex v_i, and let $\mathcal{H}_i$ be the set of all constraining half-planes corresponding to the domain subpolygons that contain the point v_i. Denote $Free_i$ to be the intersection of the complements of the half-planes in $\mathcal{H}_i$. Note, that $Free_i$ is a convex set. In the following lemma we show that $Free_i \cap R_i$ is exactly the set of points inside R_i that can attract p.

▶ **Lemma 15.** *The set of points in R_i that attract p is $Free_i \cap R_i$.*

This results in the following algorithm for computing the inverse attraction region of p. We compute the constraining half-planes of every edge of $SPT_r(p)$ of p and the corresponding domain subpolygons. Then, for every region R_i of the shortest path map of p, we compute the free region $Free_i$, where v_i is the base vertex of the region; and we add the intersection of R_i and $Free_i$ to the inverse attraction region of p. The pseudocode is presented in Algorithm 1.

Rather than computing each free space from scratch, we can compute and update free spaces using the data structure of Brodal and Jacob [7]. Their data structure allows to dynamically maintain the convex hull of a set of points and supports insertions and deletions in amortized $O(\log n)$ time using $O(n)$ space. In the dual space this is equivalent to maintaining the intersection of n half-planes. In order to achieve a total $O(n \log n)$ time, we need to provide a way to traverse recursive visibility regions and guarantee that the number of updates (insertions or deletions of half-planes) in the data structure is $O(n)$. In the rest of this section, we provide a proof for the following lemma.

Algorithm 1 Inverse attraction region.

Input: Simple polygon P, and a point $p \in P$.
Output: Inverse attraction region of p.

Compute $SPT_r(p)$ and $SPM(p)$.
for each edge $e \in SPT_r(p)$ **do**
  Compute constraining half-planes of e and corresponding domain subpolygons.
end for
for each region R_i of $SPM(p)$ with base vertex v_i **do**
  Find all the domain subpolygons that contain v_i, and compute $Free_i$.
  Intersect R with $Free_i$, and add the resulting set to the inverse attraction region of p.
end for
return Inverse attraction region of p.

▶ **Lemma 16.** *Free spaces of the recursive visibility regions can be computed in a total time of $O(n \log n)$ using $O(n)$ space.*

Proof. Consider a region R_i of $SPM(p)$ with a base vertex v_i. By Lemma 14 and Theorem 8, the set of constraining half-planes that affect the inverse attraction region inside R_i corresponds to the domain subpolygons that contain v_i.

Observe that the vertices of a domain subpolygon appear as one continuous interval along the boundary of P, as there is only one boundary segment of the subpolygon that crosses P. Then, when walking along the boundary of P, each domain subpolygon can be entered and exited at most once. All the domain polygons can be computed in $O(n \log n)$ time by shooting n rays and computing their intersection points with the boundary of P [8].

Let the vertices of P be ordered in the counter-clockwise order. For each domain subpolygon P_e, mark the two endpoints (*e.g.*, vertices v and z in Figure 6) of the boundary edge that crosses P as the first and the last vertices of P_e in accordance to the counter-clockwise order. Then, to obtain the optimal running time, we modify the second for-loop of the Algorithm 1 in the following way. Start at any vertex v_0 of P, find all the domain subpolygons that contain v_0, and initialize the dynamic convex hull data structure of Brodal and Jacob [7] with the points dual to the lines supporting the constraining half-planes of the corresponding domain subpolygons. If v_0 is a base vertex of some region R_0 of $SPM(p)$, then compute the intersection of R_0 and the free space $Free_0$) that we obtain from the dynamic convex hull data structure. Walk along the boundary of P in the counter-clockwise direction, adding to the data structure the dual points to the supporting lines of domain polygons being entered, removing from the data structure the dual points to the supporting lines of domain polygons being exited, and computing the intersection of each region of $SPM(p)$ with the free space obtained from the data structure.

The correctness of the algorithm follows from Lemma 15, and the total running time is $O(n \log n)$. Indeed, there will be $O(n)$ updates to the dynamic convex hull data structure, each requiring $O(\log n)$ amortized time. Intersecting free spaces with regions of $SPM(p)$ will take $O(n \log n)$ time in total, as the complexity of $IAR(p)$ is linear. For the pseudocode of the algorithm please refer to the full version of this paper [12]. ◀

5.1 Lower bound

The proof of the following theorem is based on a reduction from the problem of computing the lower envelope of a set of lines, which has a lower bound of $\Omega(n \log n)$ [18].

▶ **Theorem 17.** *Computing the inverse attraction region of a point in a monotone (or a simple polygon) has a lower bound of $\Omega(n \log n)$.*

Proof. Consider a set of lines L. Let l_b and l_s denote the lines in L with the biggest and smallest slope, respectively. Note that the leftmost (rightmost) edge of the lower envelope of L is part of l_b (l_s).

Without loss of generality assume that the slopes of the lines in L are positive and bounden from above by a small constant ε. We construct a monotone polygon as follows. The right part of the polygon is comprised of an axis aligned rectangle R that contains all the intersection points of the lines in L (Figure 10). Note that R can be computed in linear time. To the left of R, we construct a "zigzag" corridor in the following way. For each line l in L, in an arbitrary order, we add a corridor perpendicular to l which extends above the next arbitrarily chosen line (Figure 11). We then add a corridor with slope 1 going downward until it hits the next line. This process is continued for all lines in L.

Let the point p be the leftmost vertex of the upper chain of the corridor structure. Consider the inverse attraction region of p in the resulting monotone polygon. A point in

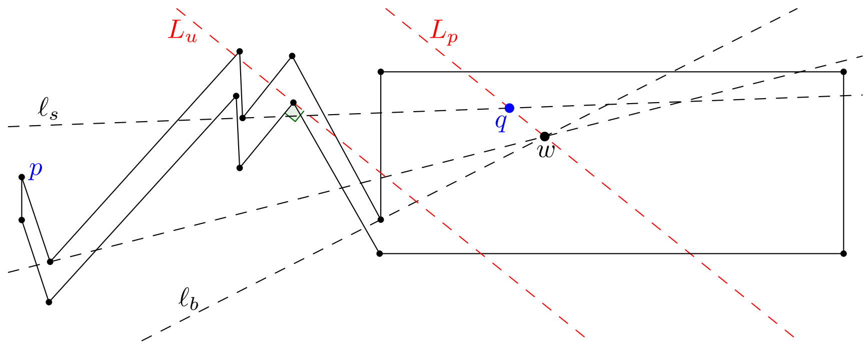

Figure 10 The final monotone polygon constructed for 3 lines.

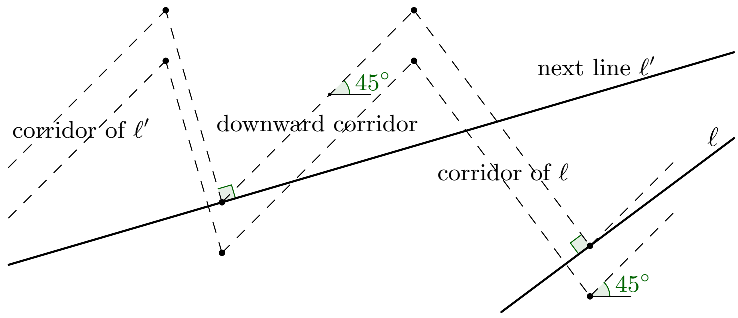

Figure 11 Adding a corridor for a line of L.

R can attract p, only if it is below all lines of L, *i.e.*, only if it is below the lower envelope of L. In addition the point needs to be above the line L_u, where L_u is the rightmost line perpendicular to a lower edge of the corridors with a slope of -1 (refer to Figure 10). In order to have all vertices of the lower envelope in the inverse attraction region, we need to guarantee that L_u is to the left of the leftmost vertex of the lower envelope, w. Let L_p be a line through w with a scope equal to -1. Let q be the intersection of L_p with l_s. We start the first corridor of the zigzag to the left of q. As the lines have similar slopes this guarantees that L_u is to the left of vertices of the lower envelope. Now it is straightforward to compute the lower envelope of L in linear time given the inverse attraction region of p. ◀

We conclude with the main result of this paper.

▶ **Theorem 18.** *The inverse attraction region of a point in a simple polygon can be computed in $\Theta(n \log n)$ time.*

References

1 Sang Won Bae, Chan-Su Shin, and Antoine Vigneron. Tight bounds for beacon-based coverage in simple rectilinear polygons. In *Proc. 12th Latin American Symposium on Theoretical Informatics*, 2016.

2 Michael Biro. *Beacon-based routing and guarding.* PhD thesis, Stony Brook University, 2013.

3 Michael Biro, Jie Gao, Justin Iwerks, Irina Kostitsyna, and Joseph S. B. Mitchell. Beacon-based routing and coverage. In *Abstr. 21st Fall Workshop on Computational Geometry*, 2011.

4 Michael Biro, Jie Gao, Justin Iwerks, Irina Kostitsyna, and Joseph S. B. Mitchell. Combinatorics of beacon-based routing and coverage. In *Proc. 25th Canadian Conference on Computational Geometry*, 2013.

5 Michael Biro, Justin Iwerks, Irina Kostitsyna, and Joseph S. B. Mitchell. Beacon-based algorithms for geometric routing. In *Proc. 13th Algorithms and Data Structures Symposium*, 2013.

6 Prosenjit Bose, Pat Morin, Ivan Stojmenović, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wirel. Netw.*, 7(6):609–616, 2001. doi:10.1023/A:1012319418150.

7 Gerth S. Brodal and Riko Jacob. Dynamic planar convex hull. In *Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

8 Bernard Chazelle and Leonidas J. Guibas. Visibility and intersection problems in plane geometry. *Discrete & Computational Geometry*, 4(6):551–581, 1989.

9 Leonidas J. Guibas, John Hershberger, Daniel Leven, Micha Sharir, and Robert Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2(1–4):209–233, 1987.

10 Brad Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, pages 243–254, New York, NY, USA, 2000. ACM. doi:10.1145/345910.345953.

11 Anne-Marie Kermarrec and Guang Tan. Greedy geographic routing in large-scale sensor networks: A minimum network decomposition approach. *IEEE/ACM Transactions on Networking*, 20:864–877, 2010.

12 Irina Kostitsyna, Bahram Kouhestani, Stefan Langerman, and David Rappaport. An optimal algorithm to compute the inverse beacon attraction region. *ArXiv e-prints*, http://arxiv.org/abs/1803.05946, 2018. arXiv:1803.05946.

13 Bahram Kouhestani, David Rappaport, and Kai Salomaa. The length of the beacon attraction trajectory. In *Proc. 27th Canadian Conference on Computational Geometry*, 2015.

14 Bahram Kouhestani, David Rappaport, and Kai Salomaa. On the inverse beacon attraction region of a point. In *Proc. 27th Canadian Conference on Computational Geometry*, 2015.

15 Bahram Kouhestani, David Rappaport, and Kai Salomaa. Routing in a polygonal terrain with the shortest beacon watchtower. *International Journal of Computational Geometry & Applications*, 68:34–47, 2018.

16 Der-Tsai Lee and Franco P. Preparata. Euclidean shortest paths in the presence of rectilinear barriers. *Networks*, 14(3):393–410, 1984.

17 Thomas Shermer. A combinatorial bound for beacon-based routing in orthogonal polygons. In *Proc. 27th Canadian Conference on Computational Geometry*, 2015.

18 Andrew C. Yao. A lower bound to finding convex hulls. *Journal of the ACM*, 28(4):780–787, 1981.