Altitude Terrain Guarding and Guarding Uni-Monotone Polygons

Ovidiu Daescu[a], Stephan Friedrichs[b,c], Hemant Malik[a],
Valentin Polishchuk[d], Christiane Schmidt[d]

[a]*Department of Computer Science, University of Texas at Dallas, {daescu, malik}@utdallas.edu*
[b]*Max Planck Institute for Informatics, Saarbrücken, Germany, sfriedri@mpi-inf.mpg.de*
[c]*Saarbrücken Graduate School of Computer Science, Saarbrücken, Germany*
[d]*Communications and Transport Systems, ITN, Linköping University, Norrköping, Sweden, {valentin.polishchuk, christiane.schmidt}@liu.se*

Abstract

We present an optimal, linear-time algorithm for the following version of terrain guarding: given a 1.5D terrain and a horizontal line, place the minimum number of guards on the line to see all of the terrain. We prove that the cardinality of the minimum guard set coincides with the cardinality of a maximum number of "witnesses", i.e., terrain points, no two of which can be seen by a single guard. We show that our results also apply to the Art Gallery problem in "monotone mountains", i.e., x-monotone polygons with a single edge as one of the boundary chains. This means that any monotone mountain is "perfect" (its guarding number is the same as its witness number); we thus establish the first non-trivial class of perfect polygons.

Keywords: Terrain Guarding Problem, Art Gallery Problem, Altitude Terrain Guarding Problem, Perfect Polygons, Monotone Polygons, Uni-monotone Polygons, Monotone Mountains

1. Introduction

Both the Art Gallery Problem (AGP) and the 1.5D Terrain Guarding Problem (TGP) are well known problems in Computational Geometry; see the classical book [1] for the former and Section 1.1 for the recent work on the latter. In the AGP, we are given a polygon P in which we have to place the minimum number of point-shaped guards, such that they see all of P.

In the 1.5D TGP, we are given an x-monotone chain of line segments in $\mathbb{R}^2$, the terrain T, on which we have to place a minimum number of point-shaped guards, such that they see T.

Both problems have been shown to be NP-hard: Krohn and Nilsson [2] proved the AGP to be hard even for monotone polygons by a reduction from MONOTONE 3SAT, and King and Krohn [3] established the NP-hardness of both the discrete and the continuous TGP (with guards restricted to the terrain vertices or guards located anywhere on the terrain) by a reduction from PLANAR 3SAT.

The problem of guarding a uni-monotone polygon (an x-monotone polygon with a single horizontal segment as one of its two chains) and the problem of guarding a terrain with guards placed on a horizontal line above the terrain appear to be problems somewhere between the 1.5D TGP and the AGP in monotone polygons. We show that, surprisingly, both problems allow for a polynomial time algorithm: a simple sweep.

Moreover, we are able to construct a maximum "witness set" (i.e., a set of points with pairwise-disjoint visibility polygons) of the same cardinality as the minimum guard set for uni-monotone polygons. Hence, we establish the first non-trivial class of "perfect polygons" [4], which are exactly the polygons in which the size of the minimum guarding set is equal to the size of the maximum witness set (the only earlier results concerned "rectilinear visibility" [5] and "staircase visibility" [4]). Since no guard can see two witness points, for any witness set W and any guard set G, $|W| \leq |G|$ holds; in particular, if we have equality, then G is a smallest-cardinality guard set (solution to the guarding problem).

One application of guarding a terrain with guards placed on a horizontal line above the terrain, the Altitude Terrain Guarding Problem (ATGP), comes from the idea of using drones to surveil a complete geographical area. Usually, these drones will not be able to fly arbitrarily high, which motivates us to cap the allowed height for guards (and without this restriction a single sufficiently high guard above the terrain will be enough). Of course, eventually we are interested in working in two dimensions and a height, the 2.5D ATGP. One dimension and height, the ATGP, is a natural starting point to develop techniques for a 2.5D ATGP. However, the 2.5D ATGP—in contrast to the 1.5D ATGP—is NP-hard by a straight-forward reduction from the (2D) AGP: we construct a terrain such that we carve out a hole for the polygon's interior and need to guard it from the altitude line at the "original" height, then we do need to find the minimum guard set for the polygon.

Roadmap. In the remainder of this section we review related work. In Section 2 we formally introduce our problems and necessary definitions, and we give some basic properties of our problems. In Section 3 we present our algorithm, prove that it computes an optimal guard set and that uni-monotone polygons are perfect; we also extend that result to monotone mountains (uni-monotone polygons in which the segment-chain is not necessarily horizontal). We show how we can obtain a runtime of $O(n^2 \log n)$; Section 3.7 shows how to find the optimal guard set in linear time (since the faster algorithm does not show the perfectness, we also keep in the slower algorithm). Finally, we conclude in Section 4.

1.1. Related work

While the TGP is quite a restricted version of the guarding problem, it is far from trivial, and understanding it is an essential step in attacking the full 2.5D terrain setting. Our work continues the line of many papers on 1.5D terrains, published during the last 10 years; below we survey some of the earlier work.

Research first focused on approximation algorithms, because NP-hardness was generally assumed, but had not been established. Ben-Moshe et al. [6] presented a first constant-factor approximation for the discrete vertex guard problem version (that is, guards may not lie anywhere on T, but are restricted to terrain vertices). This approximation algorithm constituted a building block for an O(1)-approximation of the continuous version, where guards can have arbitrary locations on T, the Continuous Terrain Guarding Problem (CTGP). Ben-Moshe et al. did not state the approximation factor, King [7] later claimed it to be a 6-approximation (with minor modifications). Clarkson and Varadarajan [8] presented a constant-factor approximation based on ε-nets and Set Cover, King [7, 9] gave a 5-approximation (first published as a 4-approximation, he corrected a flaw in the analysis in the errata). Various other, improved approximation algorithms have been presented: Elbassioni et al. [10] obtained a 4-approximation for the CGTP. Gibson et al. [11, 12], presented a Polynomial Time Approximation Scheme (PTAS) for a finite set of guard candidates. Only in 2010, after all these approximation results were published, NP-hardness of both the discrete and the continuous TGP was established by King and Krohn in the 2010 conference version of [3]. Khodakarami et al. [13] showed that the TGP is fixed-parameter tractable w.r.t. the number of layers of upper convex hulls induced by a terrain. Martinović et al. [14] proposed an approximate solver for the discrete TGP: they compute 5.5- and 6-approximations given the knowledge about pairwise visibility of the vertices as input. Friedrichs et al. [15] showed that the CTGP has a discretization of polynomial size. As the CTGP is known to be NP-hard, and Friedrichs et al. can show membership in NP, this also shows NP-completeness. And from the Polynomial Time Approximation Scheme (PTAS) for the discrete TGP from Gibson et al. [12] follows that there is a PTAS for the CTGP.

Eidenbenz [16] considered the problem of monitoring a 2.5D terrain from guards on a plane with fixed height value (which lies entirely above or partially on the terrain). He presented a logarithmic approximation for the additional restriction that each triangle in the triangulation of the terrain must be visible from only a single guard.

Hurtado et al. [17] presented algorithms for computing visibility regions in 1.5D and 2.5D terrains.

Perfect polygons were defined by Amit et al. [18] in analogy with the concept of perfect graphs (introduced by Berge [19] in the 1960s): graphs in which for every induced subgraph the clique number equals the chromatic number. The only earlier results on perfect polygons concerned so-called r-visibility (or rectangular vision) and s-visibility (or "staircase" visibility). For r-visibility two points p and q see each other if the rectangle spanned by p and q is fully contained in the polygon, for s-visibility a staircase path between p and q implies visibility. Worman and Keil [5] considered the AGP under r-visibility in orthogonal polygons and showed that these polygons are perfect under r-visibility; Motwani et al. [4] obtained similar results for s-visibility.

In his PhD Dissertation [20] Bengt Nilsson presented a linear-time algorithm to compute an optimal set of vision points on a watchman route in a *walkable polygon*, a special type of simple polygon that encompasses spiral and monotone polygons. Being developed for a more general type of polygon, rather than a uni-modal polygon, his algorithm is non-trivial and its proof of correctness and optimality is complex. In contrast, our algorithm is simple and elegant, and allows to construct a witness set of equal cardinality. In Section 3.7 we make some observations on the visibility characterizations that allow us to obtain a simple, greedy, linear-time algorithm.

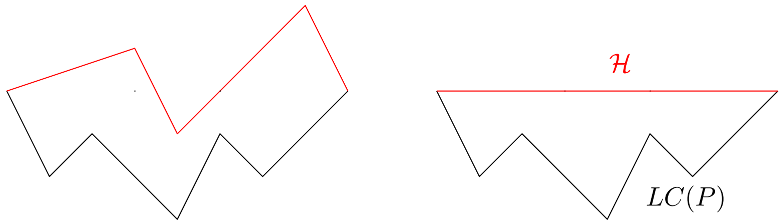


Figure 1: Left: An x-monotone polygon; the upper chain is red. Right: A uni-monotone polygon.

2. Notation, Preliminaries, and Basic Observations

In this paper we deal only with *simple* polygons, so the term "polygon" will mean "simple polygon". A polygon P is a simply-connected region whose boundary is a polygonal cycle; we assume that P is a closed set, i.e., that its boundary belongs to P. Unless specified otherwise, n will denote the number of vertices of P.

A simple polygon P is *x-monotone* (Figure 1, left) if the intersection $\ell \cap P$ of P with any vertical line ℓ is a single segment (possibly empty or consisting of just one point). It is easy to see that the boundary of a monotone polygon P decomposes into two chains between the rightmost and leftmost points of P.

Definition 1. *An x-monotone polygon P is* uni-monotone *if one of its two chains is a single horizontal segment $\mathcal{H}$ (Figure 1, right).*

W.l.o.g. we will assume that $\mathcal{H}$ is the upper chain. We denote the lower chain of P by $LC(P)$. The vertices of $LC(P)$ are denoted by $V(P) = \{v_1, \ldots, v_n\}$ from left to right, and the edges by $E(P) = \{e_1, \ldots, e_{n-1}\}$ with $e_i = \overline{v_i v_{i+1}}$.

A point $p \in P$ *sees* or *covers* $q \in P$ if $\overline{pq}$ is contained in P. Let $\mathcal{V}_P(p)$ denote the *visibility polygon* (VP) of p, i.e., $\mathcal{V}_P(p) := \{q \in P \mid p \text{ sees } q\}$. For $G \subset P$ we abbreviate $\mathcal{V}_P(G) := \bigcup_{g \in G} \mathcal{V}_P(g)$. The *Art Gallery Problem (AGP)* for P is to find a minimum-cardinality set $G \subset P$ of points (called *guards*) that collectively see all of P.

We now define the other object of our focus – terrains and altitude guarding. Say that a polygonal chain is *x-monotone* if any vertical line intersects it in at most one point.

Definition 2. *A* terrain *T is an x-monotone polygonal chain.*

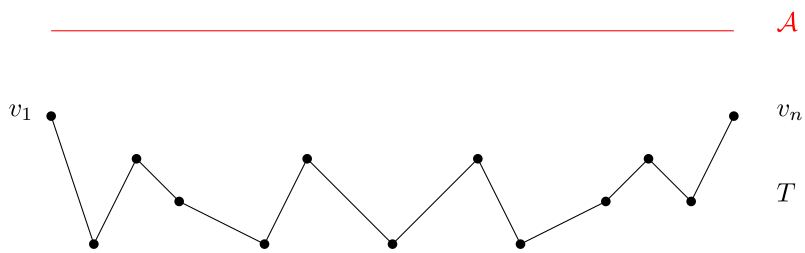


Figure 2: A terrain T in black (the vertices are the solid circles) and an altitude line $\mathcal{A}$ in red.

For instance, the lower chain $LC(P)$ of a uni-monotone polygon is a terrain. We thus reuse much of the notation for the lower chains: the vertices of T are denoted by $V(T) = \{v_1, \ldots, v_n\}$ from left to right, and the edges by $E(T) = \{e_1, \ldots, e_{n-1}\}$ where $e_i = \overline{v_i v_{i+1}}$ and $n := |V(T)|$. The *relative interior* of an edge e_i is $\text{int}(e_i) := e_i \setminus \{v_i, v_{i+1}\}$; we will say just "interior" to mean "relative interior". For two points $p, q \in T$, we write $p \leq q$ ($p < q$) if p is (strictly) left of q, i.e., has a (strictly) smaller x-coordinate.

Definition 3. *An altitude line $\mathcal{A}$ for a terrain T is a horizontal segment located above T (that is, the y-coordinate of all vertices of T is smaller than the y-coordinate of $\mathcal{A}$), with the leftmost point vertically above v_1 and the rightmost point vertically above v_n, see Figure 2.*

We adopt the same notation for points on $\mathcal{A}$ as for two points on T: for $p, q \in \mathcal{A}$, we write $p \leq q$ ($p < q$) if p is (strictly) left of q, i.e., has a (strictly) smaller x-coordinate.

A point $p \in \mathcal{A}$ *sees* or *covers* $q \in T$ if $\overline{pq}$ does not have crossing intersection with T. Let $\mathcal{V}_T(p)$ denote the *visibility region* of p, i.e., $\mathcal{V}_T(p) := \{q \in T \mid p \text{ sees } q\}$. For $G \subseteq \mathcal{A}$ we abbreviate $\mathcal{V}_T(G) := \bigcup_{g \in G} \mathcal{V}_T(g)$. We symmetrically define the *visibility region* for $q \in T$: $\mathcal{V}_T(q) := \{p \in \mathcal{A} \mid q \text{ sees } p\}$. The *Altitude Terrain Guarding Problem (ATGP)* for P is to find a minimum-cardinality set $G \subset \mathcal{A}$ of points (called *guards*) that collectively see all of T.

We now define the "strong" and "weak" visibility for *edges* of polygons and terrains:

Definition 4. *For an edge $e \in P$ or $e \in T$ the* strong visibility polygon *is the set of points that see all of e; the polygons are denoted by $\mathcal{V}^s_P(e) := \{p \in P : \forall q \in e; p \text{ sees } q\}$ and $\mathcal{V}^s_T(e) := \{p \in \mathcal{A} : \forall q \in e; p \text{ sees } q\}$. The* weak visibility polygon *of an edge e is the set of points that see at least* *one point on e; the notation is* $\mathcal{V}_P^w(e) := \{p \in P : \exists q \in e; p \text{ sees } q\}$ *and* $\mathcal{V}_T^s(e) := \{p \in \mathcal{A} : \exists q \in e; p \text{ sees } q\}$.

Last but not least, we recall definitions of witness sets and perfect polygons [18, 4].

Definition 5. *A set* $W \subset P$ *(*$W \subset T$*) is a* witness set *if* $\forall\, w_i \neq w_j \in W$ *we have* $\mathcal{V}_P(w_i) \cap \mathcal{V}_P(w_j) = \emptyset$*. A* maximum witness set W_{opt} *is a witness set of maximum cardinality,* $|W_{opt}| = \max\{|W| : \textit{witness set}\, W\}$*.*

Definition 6. *A polygon class* $\mathcal{P}$ *is* perfect *if the cardinality of an optimum guard set and the cardinality of a maximum witness set coincide for all polygons* $P \in \mathcal{P}$*.*

The following two lemmas show that for guarding uni-monotone polygons we only need guards on $\mathcal{H}$, and coverage of $LC(P)$ is sufficient to guarantee coverage of the entire polygon. Hence, the Altitude Terrain Guarding Problem (ATGP) and the Art Gallery Problem (AGP) in uni-monotone polygons are equivalent.

Lemma 1. *Let* P *be a uni-monotone polygon, with optimal guard set* G*. Then there exists a guard set* $G^{\mathcal{H}}$ *with* $|G| = |G^{\mathcal{H}}|$ *and* $g \in \mathcal{H}$ *for all* $g \in G^{\mathcal{H}}$*. That is, if we want to solve the AGP for a uni-monotone polygon, w.l.o.g. we can restrict our guards to be located on* $\mathcal{H}$*.*

Proof. Consider any optimal guard set G, let $g \in G$ be a guard not located on $\mathcal{H}$. Let $g^{\mathcal{H}}$ be the point located vertically above g on $\mathcal{H}$. Let $p \in \mathcal{V}_P(g)$ be a point seen by g. W.l.o.g. let p be located to the left of g (and $g^{\mathcal{H}}$), that is, $x(p) < x(g)$, where $x(p)$ is the x-coordinate of a point p (Figure 3). As g sees p, the segment $\overline{pg}$ does not intersect the polygon boundary, that is, the lower chain of P ($LC(P)$) is nowhere located above $\overline{pg}$: for a point $q \in LC(P)$ let $\overline{pg}(q)$ be the point on $\overline{pg}$ with the same x-coordinate as q, then $\forall q \in LC(P), x(p) \leq x(q) \leq x(g)$ we have $y(q) \leq y(\overline{pg}(q))$. Since $\overline{pg^{\mathcal{H}}}$ is above $\overline{pg}$, we have that $\overline{pg^{\mathcal{H}}}$ is also above $LC(P)$ and hence p is seen by $g^{\mathcal{H}}$ as well. That is, we have $\mathcal{V}_P(g) \subseteq \mathcal{V}_P(g^{\mathcal{H}})$, and substituting all guards with their projection on $\mathcal{H}$ does not lose coverage of any point in the polygon, while the cardinality of the guard set stays the same. □

An analogous proof shows that in the terrain guarding, we can always place guards on the altitude line $\mathcal{A}$ even if we would be allowed to place them anywhere between the terrain T and $\mathcal{A}$.

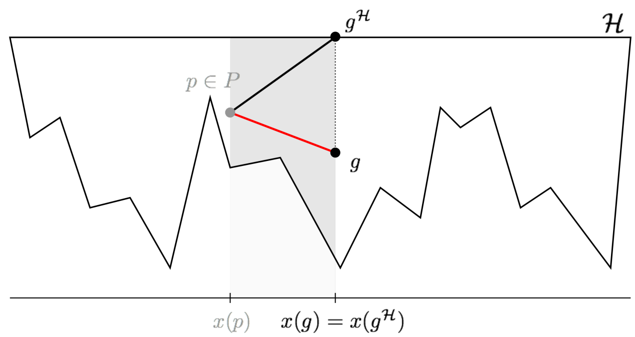


Figure 3: A uni-monotone polygon P. $g \in G$ is a guard not located on $\mathcal{H}$ and $g^{\mathcal{H}}$ is the point located vertically above g on $\mathcal{H}$. As g sees p, $g^{\mathcal{H}}$ sees p as well.

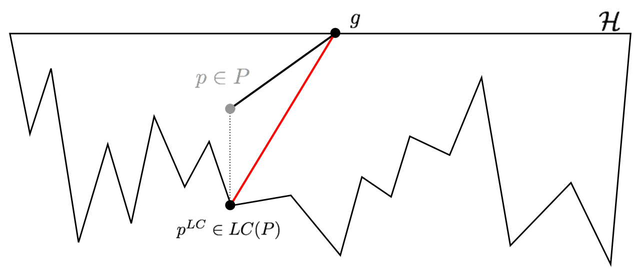


Figure 4: A uni-monotone polygon P. The guard $g \in \mathcal{G}$ sees p^{LC} the point on $LC(P)$ vertically below p. $LC(P)$ does not intersect $\overline{p^{LC}g}$ and P is uni-monotone, hence, g sees p.

Lemma 2. *Let P be a uni-monotone polygon, let $G \subset \mathcal{H}$ be a guard set that covers $LC(P)$, that is, $LC(P) \subset \mathcal{V}_P(G)$. Then G covers all of P, that is, $P \subseteq \mathcal{V}_P(G)$.*

Proof. Let $p \in P$, $p \notin LC(P)$ be a point in P. Consider the point p^{LC} that is located vertically below p on $LC(P)$. Let $g \in G$ be a guard that sees p^{LC} (as $p^{LC} \in LC(P)$ and $LC(P) \subset \mathcal{V}_P(G)$, there exists at least one such guard, possibly more than one guard in G covers p^{LC}), see Figure 4. $LC(P)$ does not intersect the line $\overline{p^{LC}g}$, and because P is uni-monotone the triangle $\Delta(g, p, p^{LC})$ is empty, hence, g sees p. $\square$

Consequently, the ATGP and the AGP for uni-monotone polygons are equivalent; we will only refer to the ATGP in the remainder of this paper, with the understanding that all our results can be applied directly to the AGP for uni-monotone polygons.

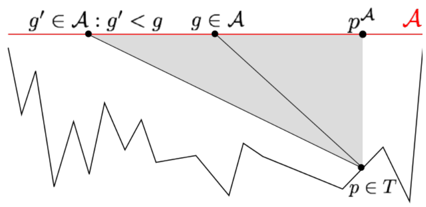


Figure 5: $p \in \mathcal{V}_T(g')$: the gray triangle $\Delta(g', p, p^{\mathcal{A}})$ is empty and so $p \in \mathcal{V}_T(g)$.

The following lemma shows a general property of guards on the altitude line, which we will use (in parts implicitly) in several cases; it essentially says that if a guard cannot see a point to its right, no guard to its left will help him by covering this point (this lemma is very much related to the well-known "order claim" [6], though the order claim holds for guards located on the terrain):

Lemma 3. *Let $g \in \mathcal{A}, p \in T, g < p$. If $p \notin \mathcal{V}_T(g)$ then $\forall g' < g, g' \in \mathcal{A} : p \notin \mathcal{V}_T(g')$.*

Proof. We show that if there exists $g' \in \mathcal{A}, g' < g$ which covers p, then g also covers p; see Figure 5 for an illustration of the proof. Since g' covers p, the segment $\overline{g'p}$ lies on or over T, and the triangle $\Delta(g', p, p^{\mathcal{A}})$, with $p^{\mathcal{A}}$ being the point located vertically above p on $\mathcal{A}$, is empty. We have $g' < g < p$, and as $x(p) = x(p^{\mathcal{A}})$ we have $g' < g < p^{\mathcal{A}}$. Hence, $\overline{gp}$ is fully contained in the triangle $\Delta(g', p, p^{\mathcal{A}})$, and lies on or over T, that is, g sees p. □

Before we present our algorithm, we conclude this section with an observation that clarifies that guarding a terrain from an altitude is intrinsically different from terrain guarding, where the guards have to be located on the terrain itself. We repeat (and extend) a definition from [15]:

Definition 7. *For a feasible guard cover C of T ($C \subset T$ for terrain guarding and $C \subset \mathcal{A}$ for terrain guarding from an altitude), an edge $e \in E$ is* critical w.r.t. *$g \in C$ if $C \setminus \{g\}$ covers some part of, but not all of the interior of e. If e is critical w.r.t. some $g \in C$, we call e a* critical edge.

That is, e is critical if and only if more than one guard is responsible for covering its interior.

$g \in C$ is a left-guard (right-guard) *of $e_i \in E$ if $g < v_i$ ($v_{i+1} < g$) and e_i is critical w.r.t. g. We call g a* left-guard (right-guard) *if it is a left-guard (right-guard) of some $e \in E$.*

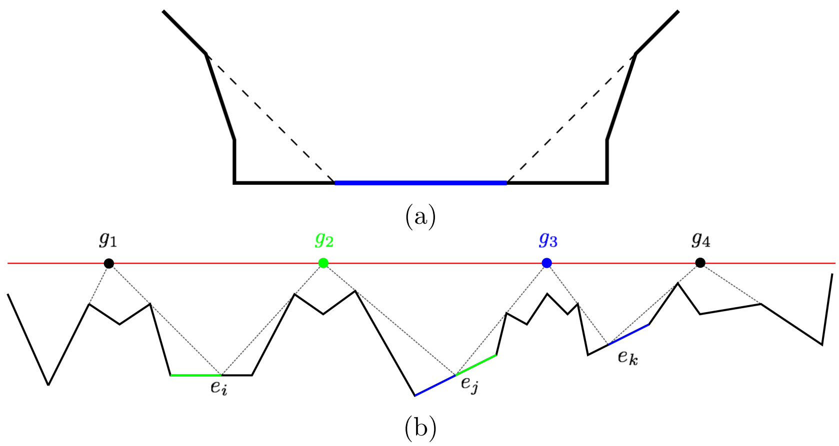


Figure 6: (a) This terrain needs two vertex- but only one non-vertex guard [6]. (b) A terrain shown in black and an altitude line $\mathcal{A}$ shown in red. Four guards, $g_1, \ldots, g_4$, of an optimal guard cover are shown as points. The green and the blue guard are both responsible for covering a critical edge both to their left and to their right: g_2 for both e_i and e_j, and g_3 for both e_j and e_k.

Observation 1. *For terrain guarding we have: any guard that is not placed on a vertex, cannot be both a left- and a right-guard [15]. (Note that a minimum guard set may need to contain guards that are not placed on vertices, see Figure 6(a).) However, for guarding a terrain from an altitude, a guard may be responsible to cover critical edges both to its left and to its right, that is, guards may be both a left- and a right-guard, see Figure 6(b).*

The observation suggests that guarding terrain from an altitude line (ATGP) could be more involved than terrain guarding (from the terrain itself), as in ATGP a guard may have to cover both left and right. However, while terrain guarding is NP-hard [3], in this paper we prove that ATGP is solvable in polynomial time.

3. Sweep Algorithm

Our algorithm is a sweep, and informally it can be described as follows: We start with an empty set of guards, $G = \emptyset$, and at the leftmost point of $\mathcal{A}$; all edges $E(T)$ are completely unseen. We sweep along $\mathcal{A}$ from left to right and place a guard g_i (and add g_i to G) whenever we could no longer see all of an edge e' if we would move more to the right. We compute the visibility polygon of g_i, $\mathcal{V}_T(g_i)$, and for each edge $e = \{v, w\}$ partially seen by g_i ($v \notin \mathcal{V}_T(g_i), w \in \mathcal{V}_T(g_i)$), we split the edge, and only keep the open interval that is not yet guarded. Thus, whenever we insert a new guard g_i we have a new set of "edges" $E_i(T)$ that are still completely unseen, and $\forall f \in E_i(T)$ we have $f \subseteq e \in E(T)$. We continue placing new guards until $T \subseteq \mathcal{V}_T(G)$. We show that there is a witness set of size $|G|$, implying that our guard set is optimal: we place a witness on e' at the point where we would lose coverage if we did not place the guard g_i.

In the remainder of this section we:

- Describe how we split partly covered edges in Subsection 3.1.
- Formalize our algorithm in Subsection 3.2.
- Prove that our guard set is optimal, and how that proves that uni-monotone polygons are perfect in Subsections 3.3 and 3.4.
- Show how that results extends to monotone mountains in Subsection 3.5.
- Show how we can efficiently preprocess our terrain, and that we obtain an algorithm runtime of $O(n^2 \log n)$ in Subsection 3.6.
- Show how we can improve the runtime to $O(n)$ in Subsection 3.7.

3.1. How to Split the Partly Seen Edges

For each edge $e \in E(T)$ in the initial set of edges we need to determine the point p_e^c that closes the interval on $\mathcal{A}$ from which all of e is visible. We denote the set of points p_e^c for all $e \in E(T)$ as the set of closing points $\mathcal{C}$, that is,

$$\mathcal{C} = \bigcup_{e \in E(T)} \{p_e^c \in \mathcal{A} : (e \subseteq \mathcal{V}_T(p_e^c)) \wedge (e \nsubseteq \mathcal{V}_T(p)\ \forall p > p_e^c,\ p \in \mathcal{A})\}.$$

The points in $\mathcal{C}$ are the rightmost points on $\mathcal{A}$ in the strong visibility polygon of the edge e, for all edges. Analogously, we define the set of opening points $\mathcal{O}$: for each edge the leftmost point p_e^o on $\mathcal{A}$, such that $e \subseteq \mathcal{V}_T(p_e^o)$,

$$\mathcal{O} = \bigcup_{e \in E(T)} \{p_e^o \in \mathcal{A} : (e \subseteq \mathcal{V}_T(p_e^o)) \wedge (e \nsubseteq \mathcal{V}_T(p)\ \forall p < p_e^o,\ p \in \mathcal{A})\}.$$

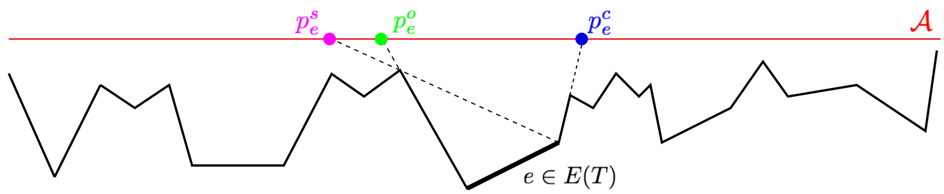


Figure 7: The closing point p_e^c, the opening point p_e^o, and the soft opening point p_e^s for an edge $e \in E(T)$. A guard to the left of p_e^s cannot see any point of e, a guard g with $p_e^s \leq g < p_e^o$ can see parts, but not all of e, a guard g with $p_e^o \leq g \leq p_e^c$ can see the complete edge e, and a guard g with $g > p_e^c$ cannot see all of e.

For each edge e the point in $\mathcal{O}$ is the leftmost point on $\mathcal{A}$ in the strong visibility polygon of e.

Moreover, whenever we place a new guard, we need to split partly seen edges to obtain the new, completely unseen, possibly open, interval, and determine the point on $\mathcal{A}$ where we would lose coverage of this edge (interval). That is, whenever we split an edge we need to add the appropriate point to $\mathcal{C}$.

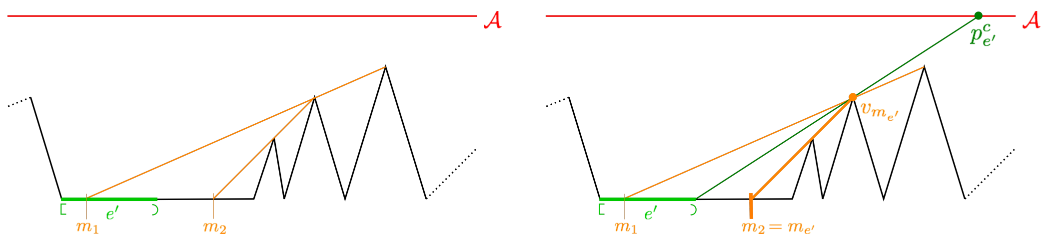
To be able to easily identify whether an edge e of the terrain needs to be split due to a new guard g, we define the set of "soft openings"

$$\mathcal{S} = \bigcup_{e \in E(T)} \{p_e^s \in \mathcal{A} : (\exists q \in e, q \in \mathcal{V}_T(p_e^s)) \wedge (\nexists q \in e, q \in \mathcal{V}_T(p) \; \forall p < p_e^s, p \in \mathcal{A})\}$$

That is, any point $p_e^s \in \mathcal{S}$ is the leftmost point on $\mathcal{A}$ of the weak visibility polygon of some edge e: if g is to the right of p_e^s (and to the left of the closing point) the guard can see at least parts of e. See Figure 7 for an illustration of the closing point, the opening point, and the soft opening point of an edge e.

So, how do we preprocess our terrain such that we can easily identify the point on $\mathcal{A}$ that we need to add to $\mathcal{C}$ when we split an edge? We make an initial sweep from the rightmost vertex to the leftmost vertex; for each vertex we shoot a ray to all other vertices to its left and mark the points, *mark points*, where these rays hit the edges of the terrain. This leaves us with $O(n^2)$ preprocessed intervals. For each mark point m we store the rightmost of the two terrain vertices that defined the ray hitting the terrain at m, let this terrain vertex be denoted by v_m. Note that for each edge $e_j = \{v_j, v_{j+1}\}$ with v_{j+1} convex vertex (seen from above the terrain), this includes v_{j+1} as a mark point.

Whenever the placement of a guard g splits an edge e such that the open interval $e' \subset e$ is not yet guarded, see for example Figure 8(a), we identify



(a) (b)

Figure 8: The terrain T is shown in black, the altitude line $\mathcal{A}$ is shown in red. The orange lines show the rays from the preprocessing step, their intersection points with the terrain define the mark points. Assume the open interval e', shown in light green, is still unseen. To identify the closing point for e' we identify the mark point to the right of e', $m_{e'}$, and shoot a ray r, shown in dark green, from the right end point of e' through $v_{m_{e'}}$. The intersection point of r and $\mathcal{A}$ defines our new closing point $p^c_{e'}$.

the first mark, $m_{e'}$ to the right of e' and shoot a ray r from the right endpoint of e' through $v_{m_{e'}}$ (the one we stored with $m_{e'}$). The intersection point of r and $\mathcal{A}$ defines our new closing point $p^c_{e'}$, see Figure 8(b).

3.2. Algorithm Pseudocode

The pseudocode for our algorithm is presented in Algorithm 1. Lines 1–3 are initialization: we start moving right from the point $a \in \mathcal{A}$ above the leftmost vertex, x_1, of the terrain (there is no guard there). Lines 5–end are the main loop of the algorithm: we repeatedly move right to the next closing point and place a guard there. The closing points are maintained in the queue $\mathcal{C}$, and an event is deleted from the queue if the new guard happens to fully see the edge (lines 10–12). The edges that are partially seen by the new guard are split into the visible and invisible parts, and the invisible part is added to the set E_g of yet-to-be-seen edges, together with the closing point for the inserted part-edge (lines 15–end).

3.3. Minimum Guard Set

Lemma 4. *The set G output by Algorithm 1 is feasible, that is, $T \subseteq \mathcal{V}_T(G)$.*

Proof. Assume there is a point $p \in T$ with $p \notin \mathcal{V}_T(G)$. For p we have $p \in e$ for some edge $e \in E(T)$. As p is not covered, there exists no guard in G in the interval $[p^o_e, p^c_e]$ on $\mathcal{A}$. Thus, p^c_e is never the event point that defines the placement of a guard in lines 6,7. Moreover, as $\nexists g_i : p^o_e \leq g_i \leq p^c_e$, e is never completely deleted from E_g in lines 10–12. Consequently, for some i we have $p_e^o > g_i$ and $g_i \geq p_e^s$ (lines 14–22). As $p \notin \mathcal{V}_T(G)$, we have $p \in e' \subset e$ (e' being the still unseen interval of e).

Again, because $p \notin \mathcal{V}_T(G)$, $\nexists g_j \in [p_e^o, p_{e'}^c] \subset \mathcal{A},\ j \geq i$. Due to line 6 no guard may be placed to the left of $p_{e'}^c$, hence, there is no guard placed in $[p_e^o, b]$ (where b is the right end point of $\mathcal{A}$). That is, e' is never deleted from E_g, a contradiction to G being the output of Algorithm 1. $\square$

To show optimality, we show that we can find a witness set W with $|W| = |G|$. We will place a witness for each guard Algorithm 1 places. First, we need an auxiliary lemmas:

Lemma 5. *Let $c \in \mathcal{C}$ be the closing point in line 6 of Algorithm 1 that enforces the placement of a guard g_i. If c is the closing point for a complete edge (and not just an edge interval), then there exists an edge $e_j = \{v_j, v_{j+1}\} \in E(T)$ for which c is the closing point, such that v_{j+1} is a reflex vertex, and v_j is a convex vertex.*

Proof. We first prove that that there exists an edge $e_j = \{v_j, v_{j+1}\} \in E(T)$ for which c is the closing point, such that v_{j+1} is a reflex vertex.

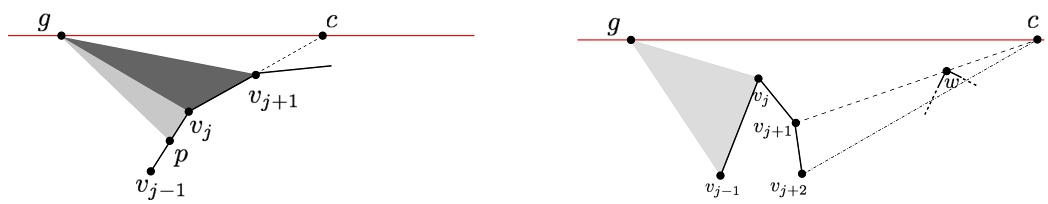
Assume there is no such edge e_j for which v_{j+1} is a reflex vertex, pick the rightmost edge e_j with v_{j+1} being a convex vertex for which c is the closing point. Let $E_c \subseteq E_g$ be the set of edges (and edge intervals) for which c is the closing point ($e_j \in E_c$). (Recall from Algorithm 1 that E_g is the set of yet-to-be-seen edges—the algorithm terminates when $E_g = \emptyset$; E_c is used only for the proof and is not part of the algorithm.) As $c = p_{e_j}^c$ is the closing point that defines the placement of a guard we have $p_e^c > c$ for all $e \in E_g \setminus E_c$ (all other active closing points are to the right of c). Because v_{j+1} sees c: $\angle(v_j, v_{j+1}, c) \leq \angle(v_j, v_{j+1}, v_{j+2}) < 180°$. We consider two cases:

- **Case 1** $\angle(v_j, v_{j+1}, c) = \angle(v_j, v_{j+1}, v_{j+2})$: In this case, c is the closing point also for e_{j+1}. Because e_j is the rightmost edge with its right vertex v_{j+1} being a convex vertex for which c is the closing point, the right vertex of e_{j+1}, v_{j+2}, must be a reflex vertex. This is a contradiction to having no such edge e_j for which the right vertex is a reflex vertex.

- **Case 2** $\angle(v_j, v_{j+1}, c) < \angle(v_j, v_{j+1}, v_{j+2})$: See Figure 9(a) for an illustration of this case. Let q be the closing point for e_{j+1}. Then the two triangles $\Delta(v_j, v_{j+1}, c)$ and $\Delta(v_{j+1}, v_{j+2}, q)$ are empty (and we have $c \geq v_{j+1}$ and $q \geq v_{j+2}$). Because T is x-monotone also the triangle

(a) (b)

Figure 9: (a) If $\angle(v_j, v_{j+1}, c) < \angle(v_j, v_{j+1}, v_{j+2})$, the triangles $\Delta(v_j, v_{j+1}, c)$, $\Delta(v_{j+1}, v_{j+2}, q)$ (shown in light gray) and the triangle $\Delta(c, q, v_{j+1})$ (shown in dark gray) are empty. Hence, c is not the closing point for e_j. (b) Placement of the witness in case c is only defined by edge intervals: we pick the rightmost such edge interval e', we have $e' = [v_j, q)$ for some point $q \in e_j, q \neq v_{j+1}$, and we place a witness at q^ε.



(a) (b)

Figure 10: Cases from the proof of Lemma 5: If v_j is a convex (a) or reflex (b) vertex of the chain g, v_j, v_{j+1}.

$\Delta(c, q, v_{j+1})$ is empty, hence, $q \in \mathcal{V}^s_T(e_j)$, a contradiction to c being e_j's closing point.

We have proved that there exists an edge $e_j = \{v_j, v_{j+1}\} \in E(T)$ for which c is the closing point, such that v_{j+1} is a reflex vertex; we now prove that v_j is a convex vertex. Assume, for the sake of contradiction, that v_j is reflex. Then c cannot be the closing point for e_{j-1}, and there exists a guard g with $g < c$ that monitors $(p, v_j] \subset e_{j-1}$; this is because irrespective of whether v_j is below or above v_{j+1}, the edge e_{j-1} is not seen by c (refer to Fig. 10). Hence, the triangle $\Delta(g, p, v_j)$ is empty. We distinguish whether the chain g, v_j, v_{j+1} has v_j as a convex or a reflex vertex.

If v_j is a convex vertex of this chain, see Figure 10(a), then also the triangle $\Delta(g, v_j, v_{j+1})$ is empty. Thus, g also monitors e_j. But if g monitors e_j, e_j would have been removed from the queue already, that is, $e_j \notin E_g$, a contradiction.

If v_j is a reflex vertex of this chain, see Figure 10(b), there has to exist a

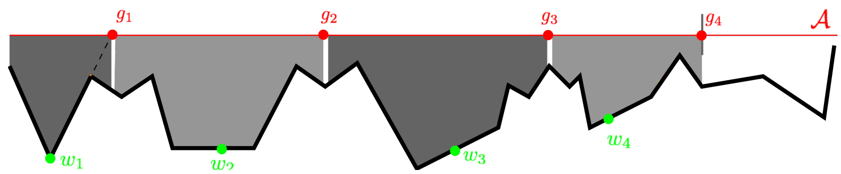


Figure 11: $S_i, i = 1, \ldots, 4$, from the proof of Lemma 6, shown in gray.

vertex w, $w > v_{j+2} > v_{j+1}$, that blocks the sight from any point to the right of c to v_{j+1} and makes c the closing point. Then all of the terrain between v_{j+1} and w lies completely below the line segment $\overline{v_{j+1}, w}$. Hence, c cannot see v_{j+2} (in fact it cannot see $(v_{j+1}, v_{j+2}] \subset e_{j+1}$). As v_j is a reflex vertex of the chain g, v_j, v_{j+1}, g cannot see v_{j+2} either. Thus, the closing point for e_{j+1} is still in the queue, and to the left of c, a contradiction to c being the closing point that is chosen in line 6 of Algorithm 1. $\square$

Now we can define our witness set:

Lemma 6. *Given the set G output by Algorithm 1, we can find a witness set W with $|W| = |G|$.*

Proof. We consider the edges or edge intervals, which define the closing point $c \in \mathcal{C}$ that leads to a placement of guard g_i in lines 6, 7 of Algorithm 1.

If c is defined by some complete edge $e_j \in E(T)$, let $E_c \subseteq E_g$ be the set of edges for which c is the closing point (we remind from Algorithm 1 that E_g is the set of yet-to-be-seen edges—the algorithm terminates when $E_g = \emptyset$). We pick the rightmost edge $e_j \in E_c$ such that v_j is a convex vertex and v_{j+1} is a reflex vertex, which exists by Lemma 5, and choose $w_i = v_j$.

Otherwise, that is, if c is only defined by edge intervals, we pick the rightmost such edge interval $e' \subset e_j$. Then $e' = [v_j, q)$ for some point $q \in e_j, q \neq v_{j+1}$, and we place a witness at q^ε, a point ε to the left of q on T: $w_i = q^\varepsilon$, see Figure 9(b).

We define $W = \{w_1, \ldots, w_{|G|}\}$. By definition $|W| = |G|$, and we still need to show that W is indeed a witness set.

Let S_i be the strip of all points with x-coordinates between $x(g_{i-1}) + \varepsilon'$ and $x(g_i)$. Let p_T be the vertical projection of a point p onto T, and $p_{\mathcal{A}}$ the vertical projection of p onto $\mathcal{A}$. $S_i = \{p \in \mathbb{R}^2 : (x(g_{i-1}) + \varepsilon' \leq x(p) \leq x(g_i)) \wedge (y(p_T) \leq y(p) \leq y(p_{\mathcal{A}}))\}$. See Figure 11 for an illustration of these strips.

We show that $\mathcal{V}_T(w_i) \subseteq S_i$ for all i, hence, $\mathcal{V}_T(w_k) \cap \mathcal{V}_T(w_\ell) = \emptyset \ \forall w_k \neq w_\ell \in W$, which shows that W is a witness set.

If $w_i = v_j$ for an edge $e_j \in E(T)$, $\mathcal{V}_T(w_i)$ contains the guard g_i, but no other guard: If g_{i-1} could see v_j, we have $\angle(g_{i-1}, v_j, v_{j+1}) \leq 180°$ because v_j is a convex vertex, thus, g_{i-1} could see all of e_j, a contradiction to $e_j \in E_g$.

Moreover, assume w_i could see some point p with $x(p) \leq x(g_{i-1})$. The terrain does not intersect the line $\overline{w_i p}$, and because the terrain is monotone the triangle $\Delta(w_i, p, g_{i-1})$ would be empty, a contradiction to g_{i-1} not seeing w_i.

If $w_i = q^\varepsilon$ for $e' = [v_j, q)$, again $\mathcal{V}_T(w_i)$ contains the guard g_i, but no other guard: If g_{i-1} could see w_i, q would not be the endpoint of the edge interval, a contradiction.

Moreover, assume w_i could see some point p with $x(p) \leq x(g_{i-1})$. Again, the terrain does not intersect the line $\overline{w_i p}$, and because the terrain is monotone the triangle $\Delta(w_i, p, g_{i-1})$ would be empty, a contradiction. □

Theorem 1. *The set G output by Algorithm 1 is optimal.*

Proof. To show that G is optimal, we need to show that G is feasible and that G is minimum, that is

$$|G| = \mathrm{OPT}(T, \mathcal{A}) := \min\{|C| \mid C \subseteq \mathcal{A} \text{ is feasible w.r.t. } \mathrm{ATGP}(T, \mathcal{A})\}.$$

Feasibility follows from Lemma 4, and by Lemma 6 we can find a witness set W with $|W| = |G|$, hence, G is minimum. □

3.4. Uni-monotone Polygons are Perfect

In the proof for Lemma 6 we showed that for the ATGP there exists a maximum witness set $W \subset T$ and a minimum guard set $G \subset \mathcal{A}$ with $|W| = |G|$. By Lemmas 1 and 2 the ATGP and the AGP for uni-monotone polygons are equivalent. Thus, also for a uni-monotone polygon P we can find a maximum witness set $W \subset LC(P) \subset P$ and a minimum guard set $G \subset \mathcal{H} \subset P$ with $|W| = |G|$. This yields:

Theorem 2. *Uni-monotone polygons are perfect.*

3.5. Guarding Monotone Mountains

We considered the Art Gallery Problem (AGP) in uni-monotone polygons, for which the upper polygonal chain is a single horizontal edge. There exist a similar definition of polygons: that of *monotone mountains* by O'Rourke [21]. A polygon P is a monotone mountain if it is a monotone polygon for which one of the two polygonal chain is a single line segment (which in contrast to a uni-monotone polygon does not have to be horizontal). By examining our argument, one can see that we never used the fact that $\mathcal{H}$ is horizontal, so all our proofs also apply to monotone mountains, and hence, we have:

Corollary 1. *Monotone mountains are perfect.*

3.6. Algorithm Runtime

Remember that we make an initial sweep from the rightmost vertex to the leftmost vertex; for each vertex we shoot a ray to all other vertices to its left and mark the points, *mark points*, where these rays hit the edges of the terrain. This leaves us with $O(n^2)$ preprocessed intervals. For each mark point m we store the rightmost of the two terrain vertices that defined the ray hitting the terrain at m, and we denote this terrain vertex by v_m.

The preprocessing step to compute the mark points costs $O(n^2 \log n)$ time by ray shooting through all pairs of vertices (this can be reduced to $O(n^2)$ with the output-sensitive algorithm for computing the visibility graph [22], which also outputs all visibility edges sorted around each vertex). Based on these we can compute the closing points for all edges of the terrain. Similarly, we compute the mark points from the left to compute the opening points (using the left vertex of an edge to shoot the ray) and the soft opening points (using the right vertex of an edge to shoot the ray).

Then, whenever we insert a guard (of which we might add $O(n)$), we need to shoot up to $O(n)$ rays through terrain vertices to the right of this guard, see Figure 12, which altogether costs $O(n^2 \log n)$ time [23]. Let the set of these rays be denoted by R_i for guard g_i. The rays may split an edge (that is, the placement of guard g_i resulted in an open interval of an edge $e' \subset e$ not yet being guarded). Let the intersection point of an edge e and a ray from R_i be denoted by r_e, it defines the right point of e'. For each of the intersection points r_e, we identify the mark point $m_{e'}$ to the right of r_e and we need to shoot a ray $\ell_{e'}$ from r_e through $v_{m'_e}$ (the terrain vertex we stored with the mark point $m_{e'}$) to compute the new closing point. That is, the intersection point of $\ell_{e'}$ and $\mathcal{A}$ defines our new closing point $p^c_{e'}$. This gives a total runtime of $O(n^2 \log n)$.

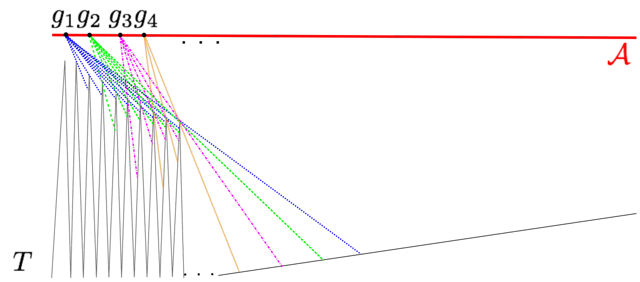


Figure 12: An example where for $O(n)$ guards each guard needs to shoot $O(n)$ (colored) rays to compute mark points to its right, yielding a lower bound of $O(n^2)$ for this approach.

3.7. Improving the Runtime

In this section we make some observations on the visibility characterizations that allow us to obtain a simple, greedy, linear-time algorithm for the ATGP (the algorithm, however, does not show the perfectness).

For a point v on T, we define the right intercept, p^c_v, and the left intercept, p^o_v, as the rightmost and leftmost point on $\mathcal{A}$ in $\mathcal{V}_P(v)$, respectively. (These are similar to the closing/opening points for edges of the terrain, defined earlier.) Equivalently, for a line segment s on T, we define the closing point, p^c_s, and the opening point, p^o_s, as the right and left intercept on $\mathcal{A}$ in $\mathcal{V}_P(s)$, respectively. For an example, consider Figure 13: x and z are the left and right intercept of point t, respectively, and w and y are the left and right intercept of point q, respectively. For the edge tq, x and y are the left and right intercept, respectively. If we move along $\mathcal{A}$, from a to b, tq becomes partially visible at w, that is, w is the soft opening point for tq, while z is the last point from which tq is partially visible. The segment is completely visible for any point on $\mathcal{A}$ between x and y. Notice that $p^o_{tq} = p^o_t$ and $p^c_{tq} = p^c_q$.

We first compute the shortest path tree from each of a and b to the vertices of T, where a and b are the endpoints of $\mathcal{A}$. This can be done in $O(n)$ time [24]. Let T_a and T_b be the shortest path trees originating from a and b, respectively. Both T_a and T_b have $O(n)$ vertices and edges. For a point $v \in T$, let $P_{v,a}$ and $P_{v,b}$ be the shortest paths from v to a and v to b, respectively. Note that these shortest paths consist of convex chains of total complexity $O(n)$.

Let $\pi_a(u)$ denote the parent of u in T_a and let $\pi_b(u)$ denote the parent of u in T_b. To find the right intercept of a vertex v of T we can extend the segment $v\pi_b(v)$ of $P_{v,b}$ and find its intersection with $\mathcal{A}$. To find the left

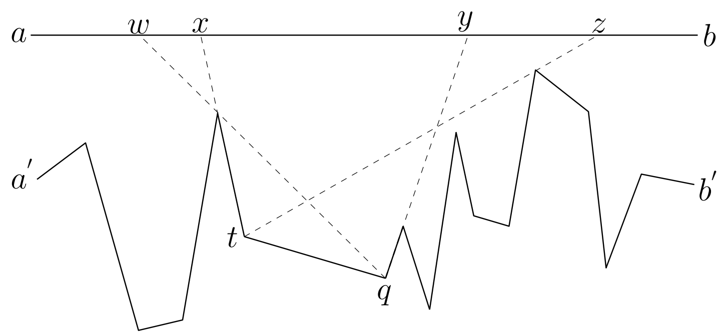


Figure 13: Terrain T (x-monotone chain from a' to b') with altitude line $\mathcal{A} = ab$. Left and right intercepts (w, x, y, and z) of points t, q and line segment tq are shown.

intercept of vertex v, we can extend the segment $v\pi_a(v)$ of $P_{v,a}$ and find its intersection with $\mathcal{A}$ (see Figure 13 and Figure 14). Similarly, we can find the left and right intercept of a line segment $s \in T$.

Our algorithm proceeds in a greedy fashion, placing guards on $\mathcal{A}$ in order, from a to b. Let $g_1, g_2, \ldots, g_i$ be the guards placed so far. As discussed in Lemma 3, all edges that lie to the left of the last placed guard, g_i, and the edge vertically below g_i, are visible by the guards placed so far. Thus, after placing g_i, we need to be concerned with the edges to the right of g_i.

Let $e = tq$ be an edge of T that lies to the right of g_i. Then tq is either (a) visible from g_i, (b) not visible from g_i (no point of tq is visible from g_i) or (c) partially visible from g_i, in which case g_i sees a sub-segment $q'q$ of tq. An easy observation from [24] and Lemma 3 is that none of the guards preceding g_i on $\mathcal{A}$ can see any point of tq'; that portion of tq' can only be seen by a guard placed to the right of g_i.

Lemma 5 shows that the guards forming the optimal set must be placed at well defined points on $\mathcal{A}$, each of which corresponds to a right intercept, p_v^c, where v is either a vertex of T or otherwise it corresponds to some point on a partially visible edge, as described earlier. This implies that, starting from g_i, the next guard will be placed at the leftmost right intercept r^l on $\mathcal{A}$, among those generated by the edges to the right of g_i. We thus walk right along the terrain, placing the guards when needed: once we reach an edge vertically below r^l we place g_{i+1} at r^l and repeat the process.

Note that to achieve linear time we cannot afford to keep the right intercepts in sorted order (see [25]). Instead, it is enough to keep track of the leftmost right intercept corresponding to the edges of T, including those generated by partially visible edges, following g_i.

Observation 2. *After placing g_{i+1} all edges of T between g_i and g_{i+1} are visible by the guards $g_1, g_2, \ldots, g_{i+1}$.*

It follows from Observation 2 that after placing g_{i+1} we do not need to be concerned with the right intercepts of the edges of T between g_i and g_{i+1}.

For a segment s of T, we define x_s^l as the x-coordinate of the leftmost point of s and x_s^r as the x-coordinate of the rightmost point of s (for an edge $s = e_i = v_i v_{i+1}$: $x_s^l = x(v_i)$ and $x_s^r = x(v_{i+1})$).

We now describe our algorithm in more details. Observe that all edges to the left of the first guard g_1 must be fully seen by g_1. To place g_1, we traverse the edges of T in order, starting with e_1. For each edge visited, we mark it as visible, compute its right intercept (its closing point) on $\mathcal{A}$, and keep track only of the leftmost such intercept, r^l. Once we reach an edge $e_i \in T$ such that $x(v_i) \leq r^l < x(v_{i+1})$ we stop, mark e_i as visible, and place g_1 at r^l. We then repeat the following inductive process. Assume guard g_i has been placed. We start with the first edge of T to the right of g_i and check if the edge is visible, not visible, or partially visible from g_i. Let e_k be the current edge. If e_k is visible then we mark it as such. If e_k is not visible then we compute its right intercept on $\mathcal{A}$ while keeping track of the leftmost right intercept, r^l, following g_i on $\mathcal{A}$. If e_k is partially visible, let e'_k be the segment of e_k not visible from g_i and let q' be the right endpoint of e'_k; we compute the right intercept of q' on $\mathcal{A}$, $p^c_{q'}$, while keeping track of r^l. Once we reach an edge $e_i \in T$ such that $x(v_i) \leq r^l < x(v_{i+1})$ we stop, mark e as visible, and place g_{i+1} at r^l. The proof that this greedy placement results in an optimal set of guards has been given in Section 3.3.

Lemma 7. *Given an edge $e = tq$ of T and a point $v \in e$, the right intercept p_v^c of v can be found in $O(1)$ amortized time. A similar claim holds for the left intercept of v.*

Proof. We present the proof for the right intercept (for the left one it is similar).

The shortest path from a and b to each vertex of T can be found in $O(n)$ time (see Subsection 3.6) and is available in the resulting shortest path tree. These shortest paths consist of convex chains. Let T_b^u be the subtree of T_b rooted at vertex u.

Recall that $\pi_b(u)$ denotes the parent of vertex u in T_b. Obviously, if v is an end vertex of e, the right intercept of v is available in constant time from

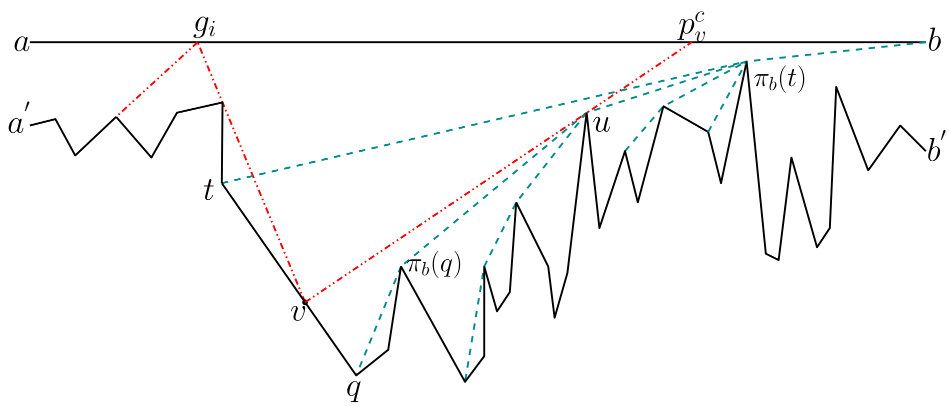


Figure 14: Line segment tq is partially seen by guard g_i. Shortest path tree originating from b is shown with dashed lines (cyan).

T_b, as the intersection of the extension of $v\pi_b(v)$ and $\mathcal{A}$. Assume v is interior to e.

To find the right intercept of v, we need to find the **first vertex** u of T_b on the shortest path, $P_{v,b}$, from v to b; the intersection of the extension of vu and $\mathcal{A}$ corresponds to p_v^c. Note that vu is tangent to a convex chain of T_b at point u, specifically the chain capturing the shortest path from q to b in T_b. Hence, we can find p_v^c by finding the tangent from v to that convex chain while traversing the chain starting at q. Moreover, the vertex u is located on the portion of the chain from q to $\pi_b(t)$. Due to the structure of the shortest paths, it is an easy observation that this subchain of T_b will not be revisited while treating an edge of T to the right of e (see Figure 14). Since the total complexity of the convex chains is $O(n)$ it follows that over all edges of T we find p_v^c in amortized $O(1)$ time. □

The visibility of an edge $e = tq$ from the last guard (g_i) placed on $\mathcal{A}$ can be found by comparing the x-coordinate of guard g_i, $x(g_i)$, with the left intercept of point q, $x(p_q^o)$, and the left intercept of point t, $x(p_t^o)$. Line segment tq is (a) completely visible from g_i if $x(p_t^o) \leq x(g_i)$, (b) not visible from g_i if $x(g_i) < x(p_q^o)$ (c) partially visible from g_i if $x(p_q^o) \leq x(g_i) < x(p_t^o)$. To find the partially visible sub-segment $q'q$ of tq we find vertex u of T_a on the shortest path from t to $\pi_a(q)$ such that the line segment ug_i joining u and g_i is tangent to the convex chain of T_a at point u. The intersection of the line supporting g_iu with tq corresponds to point q'.

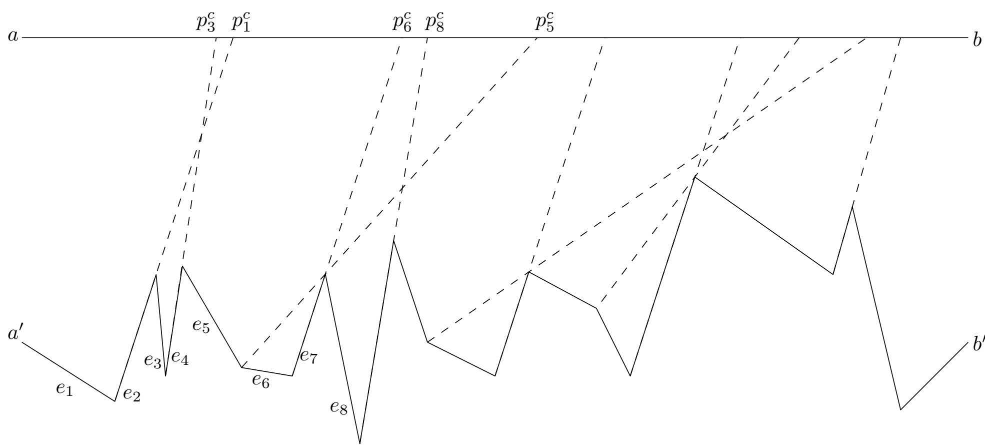


Figure 15: Terrain T with right intercept of each edge.

Lemma 8. *For an edge $e = tq$ of T that is partially visible from guard g_i the point q', defining the visible portion $q'q$ of tq from g_i, can be found in $O(1)$ amortized time.*

Proof. To find the vertex u defining the tangent $g_i u$ we traverse the convex subchain of T_a from t to u. Obviously, no other point on T to the right of q would use this subchain in a shortest path to g_i or any other point on $\mathcal{A}$ to the right of g_i. Since the total complexity of the convex chains of T_a is $O(n)$, it follows that over all edges of T we find partial visibility in amortized $O(1)$ time. □

For an example, see Figure 15. We start with e_1 and store $p^c_{e_1}$ (right intercept of e_1) as r'. We move to the next line segment, e_2, and $p^c_{e_1} = p^c_{e_2}$. For edge e_3, $p^c_{e_3} < p^c_{e_1}$, we update $\mathrm{r}' = p^c_{e_3}$. We move to the edge e_4 and $p^c_{e_3} = p^c_{e_4}$. For e_5, $p^c_{e_5} > \mathrm{r}'$, hence, no update is necessary. Moreover, $x(v_5) \leq \mathrm{r}' < x(v_6)$. Hence, we place the first guard at $r' = p^c_{e_3}$.

The algorithm visits each edge e of T only once, and the total time spent while visiting a line segment can be split into the following steps:

1. The time taken to decide the visibility of e from the last placed guard.
2. The time to find the partially visible segment of e, if needed.
3. The time to find the right intercept of a point v on edge e.
4. The time to compare p^c_e or p^c_v with r'.

Since we know the location of the last guard on $\mathcal{A}$ the first step takes constant time. The second step and the third step take $O(1)$ amortized time (see Lemma 7 and Lemma 8). The last step takes constant time. Hence, the total running time of the algorithm is $O(n)$.

Theorem 3. *The algorithm presented solves the* ATGP$(T, \mathcal{A})$ *problem in* $O(n)$ *time.*

4. Conclusion and Discussion

We presented an optimal, linear-time algorithm for guarding a 1.5D terrain from an altitude line (the ATGP) and for the art gallery problem in uni-monotone polygons and monotone mountains. We further showed that the ATGP and the AGP in uni-monotone polygons are equivalent. We proved optimality of our guard set by placing a maximum witness set (packing witnesses) of the same cardinality. Hence, we established that both uni-monotone polygons and monotone mountains are perfect.

In our algorithm, we compute the optimal guard set for a given altitude line $\mathcal{A}$. The question at which heights a_h of $\mathcal{A}$ the minimum guard set has a specified size $k \geq 1$ is open.

Moreover, while guarding a 2.5D terrain from an altitude plane above the terrain is NP-hard, it would be interesting to find approximation algorithms for that case.

Acknowledgements. We thank the anonymous reviewers for helpful comments. VP and CS are supported by Swedish Transport Administration (Trafikverket) and Swedish Research Council (Vetenskaprådet).

[1] J. O'Rourke, Art Gallery Theorems and Algorithms, International Series of Monographs on Computer Science, Oxford University Press, New York, 1987.

[2] E. Krohn, B. J. Nilsson, The complexity of guarding monotone polygons, in: Proc. of the 24th Canadian Conference on Comp. Geometry, 2012, pp. 167–172.

[3] J. King, E. Krohn, Terrain guarding is NP-hard, SIAM Journal on Computing 40 (5) (2011) 1316–1339.

[4] R. Motwani, A. Raghunathan, H. Saran, Covering orthogonal polygons with star polygons: The perfect graph approach, J. Comput. Syst. Sci. 40 (1) (1990) 19–48.

[5] C. Worman, J. M. Keil, Polygon decomposition and the orthogonal art gallery problem, Int. J. Comput. Geometry Appl. 17 (2) (2007) 105–138.

[6] B. Ben-Moshe, M. J. Katz, J. S. B. Mitchell, A constant-factor approximation algorithm for optimal 1.5D terrain guarding, SIAM Journal on Computing 36 (6) (2007) 1631–1647.

[7] J. King, A 4-approximation algorithm for guarding 1.5-dimensional terrains, in: LATIN Theoretical Informatics, 7th Latin American Symposium, 2006, pp. 629–640.

[8] K. L. Clarkson, K. R. Varadarajan, Improved approximation algorithms for geometric set cover, Discrete & Computational Geometry 37 (1) (2007) 43–58. doi:10.1007/s00454-006-1273-8.
URL `http://dx.doi.org/10.1007/s00454-006-1273-8`

[9] J. King, Errata on "a 4-approximation for guarding 1.5-dimensional terrains", `http://www.cs.mcgill.ca/~jking/papers/4apx_latin.pdf`, visited 2015-08-20.

[10] K. M. Elbassioni, E. Krohn, D. Matijevic, J. Mestre, D. Severdija, Improved approximations for guarding 1.5-dimensional terrains, Algorithmica 60 (2) (2011) 451–463.

[11] M. Gibson, G. Kanade, E. Krohn, K. Varadarajan, An approximation scheme for terrain guarding, in: I. Dinur, K. Jansen, J. Naor, J. Rolim (Eds.), Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 140–148.

[12] M. Gibson, G. Kanade, E. Krohn, K. R. Varadarajan, Guarding terrains via local search, Journal of Computational Geometry 5 (1) (2014) 168–178.
URL `http://jocg.org/index.php/jocg/article/view/128`

[13] F. Khodakarami, F. Didehvar, A. Mohades, A fixed-parameter algorithm for guarding 1.5d terrains, Theoretical Computer Science 595 (2015)

130–142. doi:10.1016/j.tcs.2015.06.028.
URL `http://dx.doi.org/10.1016/j.tcs.2015.06.028`

[14] G. Martinović, D. Matijević, D. Ševerdija, Efficient parallel implementations of approximation algorithms for guarding 1.5D terrains, Croatian Operational Research Review 6 (1) (2015) 79–89.

[15] S. Friedrichs, M. Hemmer, J. King, C. Schmidt, The continuous 1.5D terrain guarding problem: Discretization, optimal solutions, and PTAS, JoCG 7 (1) (2016) 256–284.

[16] S. Eidenbenz, Approximation algorithms for terrain guarding, Information Processing Letters 82 (2) (2002) 99–105.

[17] F. Hurtado, M. Löffler, I. Matos, V. Sacristán, M. Saumell, R. I. Silveira, F. Staals, Terrain visibility with multiple viewpoints, International Journal of Computational Geometry & Applications 24 (4) (2014) 275–306. doi:10.1142/S0218195914600085.
URL `http://dx.doi.org/10.1142/S0218195914600085`

[18] Y. Amit, J. S. Mitchell, E. Packer, Locating guards for visibility coverage of polygons, International Journal of Computational Geometry & Applications 20 (05) (2010) 601–630.

[19] C. Berge, Färbung von graphen, deren sämtliche bzw. deren ungerade kreise starr sind, Wiss. Z. Martin-Luther-Univ. Halle-Wittenberg Math.-Natur. Reihe (1961) 114115.

[20] B. Nilsson, Guarding art galleries; methods for mobile guards, Ph. D. thesis, Lund University.

[21] J. O'Rourke, Vertex π-lights for monotone mountains, in: Proc. 9th Canad. Conf. Comput. Geom., 1997, pp. 1–5.

[22] S. K. Ghosh, D. M. Mount, An output-sensitive algorithm for computing visibility graphs, SIAM Journal on Computing 20 (5) (1991) 888–910.

[23] J. Hershberger, S. Suri, A pedestrian approach to ray shooting: Shoot a ray, take a walk, Journal of Algorithms 18 (3) (1995) 403–431.

[24] D. Avis, G. T. Toussaint, An optimal algorithm for determining the visibility of a polygon from an edge, IEEE Trans. Computers 30 (12) (1981) 910–914.

[25] D. Z. Chen, O. Daescu, Maintaining visibility of a polygon with a moving point of view, Inf. Process. Lett. 65 (5) (1998) 269–275.

INPUT : Terrain T, altitude line $\mathcal{A}$, its leftmost point a, sets $\mathcal{C}, \mathcal{O}, \mathcal{S}$ of closing, opening, and soft opening points for all edges in T, all ordered from left to right.

OUTPUT: An optimal guard set G.

$E_g = E(T)$ `// set of edges that still need to be guarded`

$i := 1$

$g_0 := a$ `// the point on` $\mathcal{A}$ `before the first guard is` a, g_0 `is NOT a guard`

while $E_g \neq \emptyset$ `// as long as there are still unseen edges`

do

- 1. Move right from g_{i-1} along $\mathcal{A}$ until a closing point $c \in \mathcal{C}$ is hit
- 2. Place g_i on c, $G = G \cup \{g_i\}$, $i := i + 1$
- 3. **for** *all* $e \in E_g$ `//` $g_i \leq p_e^c$ `by construction`
  - **do**
    - **if** $p_e^o \leq g_i$ **then**
      - $E_g = E_g \setminus \{e\}$ `// if all of e is seen, delete it from` E_g
      - $\mathcal{C} = \mathcal{C} \setminus \{p_e^c\}$ `// and delete the closing point from the event queue`
    - **else**
      - **if** $p_e^s \leq g_i$ `// if` g_i `can see the right point of` e
      - **then**
        - Shoot a visibility ray from g_i onto e `// We shoot a ray from` g_i `though all vertices to the right of it, and then check if one of them is the occluding vertex, we use the ray through this occluding vertex`
        - Let the intersection point be r_e `// all points on` e `to the right of` r_e `(incl.` r_e`) are seen`
        - Identify the mark m_e immediately to the right of r_e on e
        - Shoot a ray r from r_e through v_{m_e}
        - Let $p_{e'}^c$ be the intersection point of r and $\mathcal{A}$ `//` $p_{e'}^c$ `is the closing point for the still unseen interval` $e' \subset e$
        - $\mathcal{C} = \mathcal{C} \cup \{p_{e'}^c\} \setminus \{p_e^c\}$ `// insert and delete, keeping queue sorted`
        - $E_g = E_g \cup \{e'\} \setminus \{e\}$

Algorithm 1: Optimal Guard Set for ATGP