Optimality and Competitiveness of Exploring Polygons by Mobile Robots*

Jurek Czyzowicz [†‡] Arnaud Labourel [†§] Andrzej Pelc [†¶]

July 23, 2009

Abstract

A mobile robot, represented by a point moving along a polygonal line in the plane, has to explore an unknown polygon and return to the starting point. The robot has a sensing area which can be a circle or a square centered at the robot. This area shifts while the robot moves inside the polygon, and at each point of its trajectory the robot "sees" (explores) all points for which the segment between the robot and the point is contained in the polygon and in the sensing area. We focus on two tasks: exploring the entire polygon and exploring only its boundary. We consider several scenarios: both shapes of the sensing area and the Manhattan and the Euclidean metrics.

We focus on two quality benchmarks for exploration performance: optimality (the length of the trajectory of the robot is equal to that of the optimal robot *knowing* the polygon) and competitiveness (the length of the trajectory of the robot is at most a constant multiple of that of the optimal robot knowing the polygon). Most of our results concern rectilinear polygons. We show that optimal exploration is possible in only one scenario, that of exploring the boundary by a robot with square sensing area, starting at the boundary and using the Manhattan metric. For this case we give an optimal exploration algorithm, and in all other scenarios we prove impossibility of optimal exploration. For competitiveness the situation is more optimistic: we show a competitive exploration algorithm for rectilinear polygons whenever the sensing area is a square, for both tasks, regardless of the metric and of the starting point. Finally, we show a competitive exploration algorithm for arbitrary convex polygons, for both shapes of the sensing area, regardless of the metric and of the starting point.

keywords: competitive algorithm, mobile robot, on-line exploration, polygon.

---

*A preliminary version of this paper appeared in Proc. of the 17-th Annual European Symposium on Algorithms (ESA), LNCS 5757, pp. 263–274, 2009.

[†]Département d'informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada. E-mails: jurek@uqo.ca, labourel.arnaud@gmail.com, pelc@uqo.ca

[‡]Partially supported by NSERC discovery grant.

[§]This work was done during this author's stay at the Université du Québec en Outaouais as a postdoctoral fellow.

[¶]Partially supported by NSERC discovery grant and by the Research Chair in Distributed Computing at the Université du Québec en Outaouais.

1 Introduction

The model and the problem. A mobile robot, represented by a point moving along a polygonal line in the plane, has to explore an unknown polygon and return to the starting point. We assume that the boundary is included in the polygon. The robot has a *sensing area* (abbreviated by SA in the sequel) which can be a circle or a square centered at the robot. During the exploration the robot must remain within the polygon, but its SA can partially exceed the boundaries of the polygon. At each point of its trajectory the robot "sees" (explores) all points for which the segment between the robot and the point is contained in the polygon to be explored and in the sensing area. For any explored point the robot is aware of whether this point is on the boundary of the polygon or not. We consider two tasks: exploring the entire polygon and exploring its boundary, for both shapes of the SA and for the Manhattan and the Euclidean metrics. The Manhattan metric will be called L_1 and the Euclidean metric will be called L_2. (Recall that, in the L_1-metric, the distance between two points is the sum of the differences of their coordinates.) We also differentiate the situation when the starting point of the robot is at the boundary and when it is an arbitrary point of the polygon. We assume that the robot remembers what it has explored, i.e., it keeps a partial map of the explored part of the polygon with its trajectory in it, at all times.

The quality measure of an exploration algorithm not knowing the polygon (an on-line algorithm) is the length of the trajectory of the robot, and we seek to minimize this length. We compare it to the smallest length of the trajectory of a robot *knowing* the polygon (an off-line algorithm), executing the same task (exploring the boundary or exploring the entire polygon) and starting at the same point. The ratio between these two lengths, maximized over all pairs (polygon, starting point), is the *competitive ratio* of the on-line exploration algorithm. We focus on two quality benchmarks for exploration performance: optimality (competitive ratio equal 1) and competitiveness (constant competitive ratio).

Our results. Our first set of results concerns the possibility of optimal on-line exploration. Here we consider only rectilinear polygons (those whose angles are either $\pi/2$ or $3\pi/2$). It turns out that optimal exploration is possible only in one scenario, that of exploring the boundary by a robot with square sensing area aligned with the sides of the polygon, starting at the boundary and using the L_1-metric. For this case we give an optimal exploration algorithm. In all other scenarios (when *either* the entire polygon has to be explored, *or* the sensing area is a circle, *or* the metric is L_2, *or* the starting point may be strictly inside the polygon) we prove impossibility of optimal on-line exploration.

For competitiveness, the situation is more optimistic: our optimal boundary exploration algorithm yields a competitive exploration algorithm for rectilinear polygons whenever the sensing area is a square aligned with the sides of the polygon, for both tasks (exploring the boundary or the entire polygon) regardless of the metric and of the starting point. Finally, we show a competitive exploration algorithm for arbitrary convex polygons, for both shapes of the sensing area, regardless of the metric and of the starting point.

To the best of our knowledge we propose the first competitive on-line algorithm to explore arbitrary rectilinear polygons with some limited sensing area.

Related work. Exploration of unknown environments by mobile robots was extensively studied in the literature under many different models. One of the most important works in this domain is [5] where the sensing area is unlimited. The authors gave a 2-competitive algorithm for rectilinear polygon exploration. The competitive ratio was later improved to 5/3 in [8]. It was shown in [13] that there is no deterministic algorithm for this problem better than 5/4-competitive and that there

exists a 5/4-competitive randomized algorithm solving it. All these results hold for the L_1-metric. Upper bounds for the L_2-metric can be obtained from the fact that any α-competitive algorithm for the L_1-metric is $\alpha\sqrt{2}$-competitive for the L_2-metric [5]. The case of non-rectilinear polygons was also studied in [4, 10] and a competitive algorithm was given in this case.

For polygonal environments with an arbitrary number of polygonal obstacles, it was shown in [5] that no competitive strategy exists, even if all obstacles are parallelograms. Later, this result was improved in [1] by giving a lower bound in $\Omega(\sqrt{k})$ for the competitive ratio of any on-line algorithm exploring a polygon with k obstacles. This bound remains true even for rectangular obstacles. Nevertheless, if the number of obstacles is bounded by a constant m, then there exists a competitive algorithm with competitive ratio in $O(m)$ [4].

Exploration by a robot with a limited sensing area has been studied, e.g., in [6, 7, 11, 12, 15]. This model is interesting to study, since it is justified by real world constraints. Indeed, computer vision algorithms based on information obtained by sensors, such as stereo or structured-light finder, can reliably compute visibility scenes only up to a limited range [7]. To the best of our knowledge, there were no previous results concerning competitive on-line exploration for *arbitrary* rectilinear polygons with limited visibility.

The off-line exploration problem with limited SA is related to older problems such as *lawn mowing*, *pocket milling* and *ice rink* problems. All these three problems are concerned with finding an optimal path of a tool moving on a surface (grass area to mow, pocket to mill or ice rink to sweep), such that all points of the surface are covered by the tool (a mower, cutter or ice rink machine) at least once during its travel. The only difference between exploration and the lawn mowing problem is that the robot is not allowed to leave the environment, while the mower can exit the surface. The ice rink problem is the same as the lawn mowing problem, except for the notion of the optimal path. In lawn mowing, only the length of the path is considered, while in the ice rink problem we also need to take into account the number of turns done by the robot, since those turns are costly [14]. In the pocket milling problem, not only the robot cannot leave the surface but also the cutter must not leave it. Here, the goal is to find a shortest path that covers the maximum area possible. The first two problems are NP-hard and the complexity of the third one is unknown [9]. All three problems admit polynomial time approximation algorithms [2, 14].

On-line exploration with limited SA has been studied, e.g., in [6, 11, 12]. Unlike in our model, the robot in [6] can see slightly farther than its tool (six times the tool range). The authors describe an on-line algorithm with competitive ratio $1 + 3(\Pi D/A)$, where Π is a quantity depending on the perimeter of P, D the size of the tool and A the area of P. Since the ratio $\Pi D/A$ can be arbitrarily large, their algorithm is *not* competitive in the general case. Moreover, the exploration in [6] fails on a certain type of polygons, such as those with narrow corridors.

In [11, 12], the authors consider the exploration of a particular class of polygons: those composed of complete identical squares, called cells of size a priori known to the robot. In this model, the robot explores all points in a cell when it enters the cell for the first time, and can move in one step to any adjacent cell. The cost of the exploration is measured by the number of steps. There exists a 2-competitive algorithm for exploration of such polygons with obstacles [11]. For polygons without obstacles, there exists a 4/3-competitive algorithm for exploration and no algorithm can achieve a competitive ratio better than 7/6 [12].

There are only a few papers on how to explore the boundary of a terrain with limited sensing area. This problem was first considered in [15] (in its off-line version) using a reduction to the safari route problem. The *safari route problem* consists in finding a shortest trajectory, starting at the

point s of the boundary of a polygon P and going back to s, that visits a specified set of polygons $\mathcal{P}$ contained in P. It is assumed in [15] that the polygons in $\mathcal{P}$ are attached to the boundary of P (share at least one point with the boundary of P), since otherwise the problem is NP-hard [15]. The author gives a $O(mn^2)$ algorithm solving this problem, where m is the cardinality of $\mathcal{P}$ and n is the total number of vertices of P and polygons in $\mathcal{P}$. It is shown that an optimal safari route visiting all the circular sectors of vertices corresponding to the angles of P, (i.e., the region inside P from which the vertex is visible), is an optimal boundary exploration trajectory [15]. To solve the safari route problem, circular sectors are approximated with polygons and the obtained solution is within 0.3% of optimal. It is computed in cubic time.

2 Definitions and preliminary results

In this section and in the part of the paper concerning optimality of exploration, we only consider rectilinear polygons. Let P be such a polygon. For convenience, without loss of generality, we assume that each side of the polygon P is either parallel to the x-axis (east-west sides) or to the y-axis (north-south sides).

A *rectilinear* trajectory path has each of its segments parallel to either the x-axis or the y-axis. Since in the L_1-metric there is always a rectilinear path among the shortest paths between two points, we consider only rectilinear paths and we drop the word "rectilinear" in all considerations regarding the L_1-metric. In particular, we use this convention in this section and in Section 3.1.

A segment T contained in a polygon P is *separating*, if it divides P into two simple polygons called the *subpolygons defined by* T. The *foreign polygon* defined by T according to a point u, denoted by $FP_u(T)$, is the subpolygon not containing u. Note that the foreign polygon is undefined if $u \in T$. A separating segment T *dominates* a separating segment T' according to the point u, if $FP_u(T)$ is strictly contained in $FP_u(T')$. For instance, in Fig. 1(a), the segment T_1 is dominated by segment V according to point r_0.

The robot at position r *explores* a point x, if the segment $\overline{rx}$ is included both in the polygon and in the SA centered in r. We consider two types of SA: a *round* SA which is a disc of diameter 2 and a *square* SA which is a 2×2 square. For exploration of rectilinear polygons, we assume that the sides of a square SA are aligned with the sides of the polygons. An *exploration trajectory* of polygon P is a path contained in P such that each point of P is explored by the robot at some point of this path. A *boundary exploration trajectory* is a trajectory of a robot inside the polygon P, exploring the boundary of P. In both cases, the start and the end of the trajectory are equal and are denoted by r_0.

In our proofs we use the following results from the literature.

Proposition 2.1 (Corollary 2.6 in [5]) *Let M be a separating segment of a rectilinear polygon P. Let u be any point in P and v be any point in M. There is a shortest path from u to v, which consists of a shortest path from u to M, meeting M at point t, followed by the segment $\overline{tv} \subseteq M$.*

Proposition 2.2 (Lemma 2.5 in [5]) *Let u be a point and let M and M' be two separating segments, such that M' dominates M according to u. There is a shortest path from u to M', which consists of a shortest path from u to M, meeting M at point t, followed by a path from t to M'.*

Proposition 2.3 (Proposition 2.7 in [5]) *Let M and M' be two intersecting north-south and east-west separating segments of a polygon P, dividing P into four subpolygons such that points u*

and v are in diagonally opposite quadrants. (They may be on M and/or M'). There is a shortest path from u to v, which consists of a path from u to the intersection $X(M, M')$ followed by a path from $X(M, M')$ to v.

Proposition 2.4 (Corollary 2.8 in [5]) *Let M and M' be two intersecting north-south and east-west separating segments of a polygon P, dividing P into four subpolygons such that points u and v are on the same side of M' but on opposite sides of M. Among all paths from u to v that visit M', there is a shortest one which consists of a path from u to the intersection $X(M, M')$, followed by a path from $X(M, M')$ to v.*

Proposition 2.5 (Lemma 3 in [3]) *For any boundary exploration trajectory with crossings in a polygon P, there exists a boundary exploration trajectory with no crossings that uses the same trajectory parts between crossings, but in different order.*

For each side S of a polygon P, we extend S inside P, possibly from both ends, until it first hits the boundary of P. Each contiguous section of the resulting segment, if any, excluding S itself, is called an *extension segment* (cf. [5]) associated with S (see Fig. 1(a)). For each side S of a polygon P, we draw the line L parallel to S at distance one from it, on the side of the interior of P. If this line intersects P, we define the *vicinity segment* associated with S, as the part of L between the closest point of $P \cap L$ from S in clockwise order along the boundary and the closest one in anti-clockwise order (see Fig. 1(a)).

Each extension or vicinity segment M of side S is a separating segment. In the rest of the paper, any domination relation or foreign polygon $FP(M)$ is defined according to point r_0, if no other point of reference is specified. If $r_0 \in M$, we set $FP(M)$ to be the subpolygon defined by M that contains S. Starting at r_0, if side $S \in FP(M)$, where M is an extension or vicinity segment of S, then S can become explored only if M is visited (i.e., either crossed or touched). If this is the case, we call M a *necessary segment* of S. For instance, the segment T_1 in Fig. 1(a) is necessary. For two necessary (extension or vicinity) segments M_1 and M_2, if M_1 dominates M_2 then there is no way to visit M_1 without crossing M_2 from r_0. So, we can ignore M_2, since it is automatically visited, if we visit M_1. A non-dominated necessary segment is called *essential*. To see all sides of a polygon, starting at r_0, the robot has to visit every essential segment.

If the starting point r_0 is on the boundary of P, then it induces a *natural order* of essential segments, clockwise along the boundary of the polygon P: $E_1, E_2, \ldots, E_m$, where E_1 is the first essential segment encountered when moving clockwise along the boundary from r_0, and so on. For $i \in 1, \ldots, m$, we denote by x_i the point on E_i at the minimum distance from point x_{i-1}, with the starting point $r_0 = x_0$. As shown in [5], these points are uniquely defined by r_0. This trajectory from x_0 to x_m, and back directly from x_m to x_0, is called GE for 'Greedy Essential'. See Fig. 1(b) for an example of such a trajectory in a polygon.

We define, similarly as in [5], a new "reduced" polygon P' obtained from P as follows: For each essential segment E, remove from P the foreign subpolygon defined by E. The following lemma is used in the proofs of Lemma 2.2 and Theorem 4.1.

Lemma 2.1 *Each optimal boundary exploration trajectory for P is entirely contained in the reduced polygon P'.*

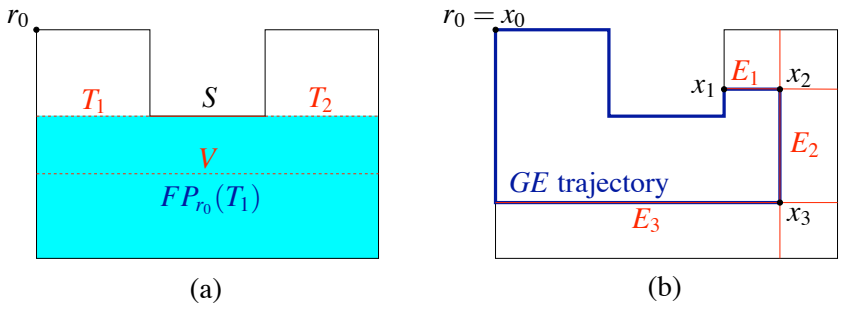

Figure 1: (a) Vicinity segment V and extension segments T_1 and T_2 associated with side S. (b) Example of a GE trajectory.

Proof: Assume, for contradiction, that there is an optimal trajectory T for boundary exploration of P that is not contained in P'. So, there is an essential segment E, such that T crosses E and visits its foreign polygon. We can assume, without loss of generality, that E is a north-south segment.

Consider the trajectory T' obtained by replacing the part s of T contained in $FP(E)$ by its orthogonal projection s' on the line L defined by E. The trajectory T' is strictly shorter than T, since its part s' is strictly shorter than s.

Now, we show that T' is a boundary exploration trajectory shorter than T which is assumed to be optimal, a contradiction. To show that T' is a boundary exploration trajectory, it suffices to show that the polygon $FP(E)$ is explored by trajectory s'. Consider three points: $u \in FP(E), v \in s, v' \in s'$, such that u is explored from v and v' is the orthogonal projection of v onto L. Since u is explored from v, the difference between their y-coordinates is at most one. By the projection on L, the y-coordinate is preserved and the difference between the y-coordinates of u and v' is at most one. The difference between x-coordinates of u and v' is also at most one. Indeed, any larger difference would imply the existence of a necessary vicinity segment parallel to E contained in $FP(E)$ and would contradict the fact that E is essential. Hence, the point u is in the (square) SA centered in v'.

To show that u is explored from v', it remains to show that the segment $\overline{uv'}$ is inside P. Assume, for contradiction, that $\overline{uv'}$ is not inside P. Since $\overline{uv}$ is inside P, the only possibility is that there is a $3\pi/2$-angle b, as depicted in Fig. 2. The side incident to b not explored from v' has an extension segment E' parallel to E and inside $FP(E)$. E' is necessary and dominates E, a contradiction with the fact that E is essential. So, uv' is inside P and u is explored by v'. Any point of $FP(E)$ explored by s is explored by s', and so T is a boundary exploration trajectory.

□

Lemma 2.2 *Any boundary exploration trajectory is not shorter than GE.*

Proof: We first observe that there always exists a shortest path visiting the essential extensions in the clockwise order, that does not self-intersect. This is a slight variation of a result from [3], with extension and vicinity essential segments replacing essential extension segments. The result still holds by Proposition 2.5 and Lemma 2.1.

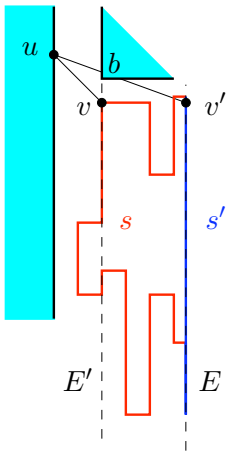

Figure 2: Example of trajectories s and s'

Hence, GE is an optimal way of visiting each essential segment in clockwise order [5]. Since any boundary exploration trajectory has to visit all the essential segments, any boundary exploration trajectory starting from and ending at r_0 is not shorter than GE. □

3 Optimality

3.1 The optimal boundary exploration algorithm

In this section, we assume that the SA is a 2×2 square aligned with sides of a rectilinear polygon. Our aim is to construct a boundary exploration algorithm starting at a boundary point r_0 and following GE as closely as possible. Unfortunately, in the case of a robot with bounded SA (unlike the robot from [5] which had unbounded visibility), it is impossible for an on-line algorithm to visit essential extensions greedily, using shortest paths. The following proposition shows this significant difference between our scenario and that from [5].

Proposition 3.1 *There is no on-line algorithm that greedily visits the essential segments of every polygon, i.e., that visits the essential segments by following shortest paths between them, even starting at the boundary.*

Proof: We consider two polygons P and P' depicted in Fig. 3. Let the corner x be the starting point of the robot. Since both polygons P and P' are symmetric, we can assume without loss of generality that the first essential segment visited is E_1 in P and is E_1' in P'. In order to achieve a shortest path from x to either E_1 or E_1', the robot must move along the side $\overline{xy}$. Notice that both polygons P and P' look the same to the robot when it moves along the side $\overline{xy}$. So, the adversary can arbitrarily choose one of the two polygons when the robot stops moving along the side $\overline{xy}$. The adversarial strategy consists in taking the polygon P', if the robot stops moving along $\overline{xy}$ at distance at least one from y, and in taking the polygon P otherwise. In the first case, the trajectory of the robot from x to E_1' is not a shortest path, since the robot does not move along

$\overline{xy}$ to visit E_1'. In the second case, the trajectory of the robot from z (intersection of E_1 and $\overline{xy}$) to E_2 is not a shortest path, since the robot does not move along E_1 to visit E_2. Hence, in both cases, the robot does not greedily visit the essential segments of the polygon.

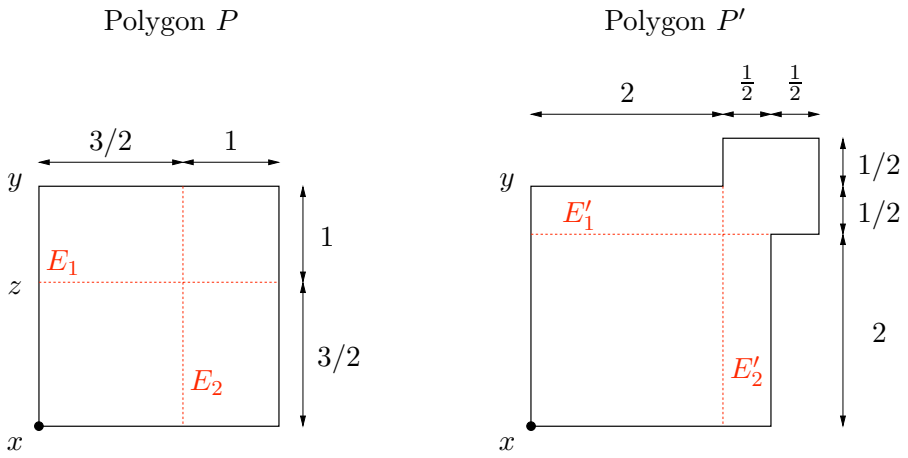

Figure 3: The polygons P and P'

□

Since, as shown above, our bounded visibility scenario is more difficult than that from [5], our optimal boundary exploration algorithm must also be more subtle. Its idea is as follows.

The robot tries to increase the contiguous part of the boundary seen to date. The rest of the boundary is not yet explored by the robot for three possible reasons: an obstructing angle limiting the view of the currently explored side, a $3\pi/2$-angle terminating the currently explored side and obstructing the view of the next side, or finally the end of the SA limiting the view of the currently explored side. The strategy of the robot is to move towards the extension corresponding to the obstructing angle (in the first two cases) and to move parallel to the currently explored side (in the third case). Due to limited visibility, no necessary segment is seen by the robot in the third case, which is a crucial difference between our scenario and that from [5]. While it is impossible to move between consecutive essential segments using shortest paths, we prove that for every essential segment there is some essential segment following it (not necessarily the next one) which the robot reaches by a shortest of all paths visiting the intermediate essential segments. Proving this property is the crucial and technically most difficult part of the algorithm analysis.

Algorithm `BOUNDARY-ON-LINE-EXPLORATION` (BOE, for short)
INPUT: A starting point r_0 on the boundary of the polygon to be explored.
OUTPUT: A shortest boundary exploration trajectory, starting and ending at r_0.

We denote by C the contiguous part of the boundary, starting clockwise from r_0, that has been explored so far by the robot, and we call *frontier*, denoted by f, the end of C. The current position of the robot is denoted by r.

Repeat the following strategy until C becomes the boundary of a simple polygon, updating r, f and C whenever any change occurs.

Case 1: *There is an obstructing angle b, i.e., r, b and f are aligned and b is a $3\pi/2$ angle not in C (see Fig. 4(a))*
Move towards the extension $E(b)$ of the side $U(b)$ incident to b and not explored from r. The strategy used to reach $E(b)$ is to move parallel to the other side $S(b)$ incident to b whenever possible, and move towards $S(b)$, parallel to $E(b)$, until it becomes possible again to move parallel to $S(b)$, otherwise.
Case 2: *f is a $3\pi/2$ angle and r is not on the extension $E(f)$ of the side $U(f)$ incident to f and not explored from r. (see Fig. 4(b))*
Same as Case 1 with f instead of b.
Case 3: *There is no obstructing angle, and either f is a $3\pi/2$ angle and r is on the extension $E(f)$ of the side $U(f)$ incident to f and not explored from r, or f is not a $3\pi/2$ angle. (see Fig. 4(c))*
If f is a $3\pi/2$ angle then $S(f) = U(f)$, otherwise $S(f)$ is the side containing f. Move parallel to the side $S(f)$ towards f until:

Case 3.1: *Condition of Case 1 occurs*
Follow Case 1.
Case 3.2: *Condition of Case 2 occurs*
Follow Case 2.
Case 3.3: *A new $\pi/2$ angle a is explored and belongs to C (the robot reaches the vicinity segment $V(a)$ of the new side $U(a)$ incident to a)*
Do nothing (the algorithm proceeds to the next iteration of the repeat loop).

When the above **Repeat** loop is completed (C is a simple polygon), follow a shortest path to r_0 and stop.

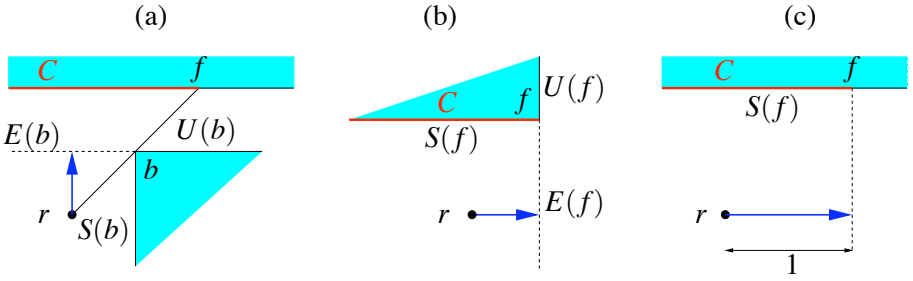

Figure 4: The three possible configurations during the execution of Algorithm BOE

In Fig. 5 we show an example of the execution of Algorithm BOE. The robot starts at r_0 where Case 3 occurs. It goes North until it sees a $\pi/2$ angle at point r_1, it recognizes Case 3.3 and starts a new iteration of the Repeat loop with Case 2. It goes East until point r_2. Case 3 occurs. The robot goes North until it sees a $\pi/2$ angle at point r_3. It recognizes Case 3.3 and starts a new iteration with Case 3. It goes East until point x, where its vision is obstructed by angle y. Case 3.1 occurs. The robot goes North until point r_4. Subsequently the robot continues the exploration in three iterations of the Repeat loop, remaining in Case 3.3 in points r_5, r_6, r_7. At r_7 it has explored the entire boundary and returns to r_0 by a shortest rectilinear path.

The main result of this section is that Algorithm BOE is optimal.

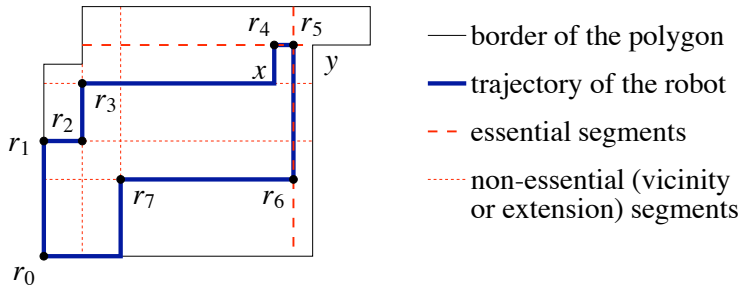

Figure 5: Example of an execution of Algorithm BOE on a polygon

Theorem 3.1 *Algorithm* `BOUNDARY-ON-LINE-EXPLORATION` *is an optimal on-line algorithm for the boundary exploration of rectilinear polygons with square SA in the L_1-metric, starting and ending at a point of the boundary.*

First, we show that Algorithm BOE eventually terminates.

Lemma 3.1 *Algorithm BOE eventually terminates with C set to the boundary of the input polygon P.*

Proof: Let n be the number of sides of P. Let $S = \overline{de}$ be the the side of P currently explored, i.e., the side which partially belongs to C. We show that after at most $O(n)$ iterations of the main loop of BOE, starting from any point, the side S will be entirely in C. For the sake of uniformity, we consider the side $\overline{ab}$ containing r_0 as two sides $\overline{ar_0}$ and $\overline{r_0b}$.

The first end-point d of S in clockwise order from r_0 is in C. Two cases are possible.

Case A: d is the only point of S in C.

In this case d must be a $3\pi/2$ angle. The algorithm follows the strategy for Case 2 (one iteration) and after the next iteration some other points of S are added to C and Case B occurs.

Case B: there is a part of S of length non-zero in C.

If the next iteration corresponds to Cases 2, 3.2, or 3.3, then after this iteration the angle e is added to C and hence the entire side S is explored. The remaining cases (i.e., Case 1 and Case 3.1), can occur at most n times until the entire side S is added to C, since a single angle cannot obstruct the vision twice during the exploration of a side.

Hence the algorithm eventually terminates. □

Let l be the number of iterations of the main loop of Algorithm BOE before terminating. For $i = 1, 2, \ldots, l$, the robot is at point r_i at the end of the i-th iteration of the main loop. The point r_i is either on a vicinity or on an extension segment denoted by M_i. Indeed, at the end of an iteration corresponding to Cases 1 or 3.1, the robot is on the extension segment $E(b)$ of side $U(b)$. For Cases 2 or 3.2, the robot is on the extension segment $E(f)$ of side $U(f)$. Finally, for Case 3.3, the robot is on the vicinity segment $V(a)$ of side $U(a)$.

We define a new trajectory BOE' that reaches segments M_i in a greedy way. For $i \in 1, \ldots, l$, we denote by z_i the point on M_i at the minimum distance (in the L_1-metric) from point z_{i-1}, with $r_0 = z_0 = z_{l+1}$. More formally, the trajectory BOE' is the one following a shortest path from z_{i-1} to z_i, for all $1 \leq i \leq l+1$.

Although BOE might not follow a shortest path between the segment M_i and M_{i+1} for some i, its total length turns out to be equal to that of BOE'. We denote by $BOE[r_i, r_k]$ (resp. $BOE'[z_i, z_k]$) the part of the trajectory BOE (resp. BOE') between the points r_i and r_k (resp. z_i and z_k).

Lemma 3.2 *The BOE' trajectory has the same length as the BOE trajectory.*

Proof: We show that for all i, there exists a j, such that $BOE[r_i, r_{i+j}]$ is a shortest path from point r_i to M_{i+j} that visits segments M_{i+k} for $1 \leq k < j$. The proof depends on the type of the $(i+1)$-th iteration of Algorithm BOE.
Case 1: The robot follows a shortest path from r_i to the extension segment $E(b) = M_{i+1}$ as shown in [5]. Hence, the property holds for $j = 1$.
Case 2: Same as Case 1 with f instead of b.
Case 3.1: The robot moves parallel to $S(f)$ and then moves towards the extension segment $E(b)$, where b obstructs the vision to $S(f) = \overline{dv}$ (with d the first vertex of $S(f)$ in clockwise order) from the robot. Assume, without loss of generality, that the robot moves east when moving parallel to $S(f)$ ($S(f)$ is an east-west side) and is south of $S(f)$.

In order to explore the vertex v, the robot has to execute an iteration corresponding to Cases 2, 3.2 or 3.3. Let j denote the number of iterations executed by the robot to fully explore $S(f)$, the last one corresponding to Case 2, 3.2 or 3.3, needed to explore v.

Let S' be the side following the side $S(f)$ in the clockwise order. The segment M_{i+j} is either the extension segment of S', if the angle between $S(f)$ and S' is a $3\pi/2$ angle, or the vicinity segment of S', if the angle between $S(f)$ and S' is a $\pi/2$ angle. In both cases, M_{i+j} is perpendicular to all M_{i+k} for $0 \leq k < j$ and is east of point r_{i+1}.

During the iterations corresponding to Cases 1 or 3.1, the robot moves either north or east, since for all $1 \leq k < j$, M_{i+k} is an east-west segment and the obstructing angle b_k is in the north-east quadrant of the SA of the robot. During the last iteration corresponding to Cases 2, 3.2, or 3.3, the robot moves either north or east, since M_{i+j} is a north-south segment and the angle f (Cases 2 or 3.2) or the angle a (Case 3.3) is in the north-east quadrant of the SA of the robot. Hence, the path from r_i to r_{i+j} is a shortest path.

We show that the point r_{i+j} is the point of M_{i+j} at minimal distance from r_i. Indeed, it is reached by minimal x-axis and y-axis shifting, since the path is monotone and the robot moves north only until reaching the y-coordinate of the angle b_{j-1}. Hence, the path is a shortest path to M_{i+j} visiting all M_{i+k} for $1 \leq k < j$, and the property is verified.
Case 3.2: The robot moves parallel to $S(f)$ and then applies the strategy of Case 2. Since this strategy consists in moving parallel to $S(f)$ whenever it is possible, the property is verified, as in Case 2.
Case 3.3: The robot moves parallel to $S(f)$ from r_i to the vicinity segment M_{i+1} of the side immediately after $S(f)$ in clockwise order. The path followed by the robot to reach M_{i+1} is a shortest path, since $S(f)$ is perpendicular to M_{i+1}. Hence, the property holds for $j = 1$.

Recall that $r_0 = z_0$. We showed that $r_i = z_i$ implies $|\,BOE[r_i, r_{i+j}]| = |\,BOE'[z_i, z_{i+j}]|$, and $r_{i+j} = z_{i+j}$, for the index j (depending on i) determined above, since $BOE[r_i, r_{i+j}]$ is a shortest

path from r_i to M_{i+j}. It follows by induction that $|\,BOE\,| = |\,BOE'\,|$. □

Lemma 3.3 *Every essential segment is in the set* $\{M_1, M_2, \ldots, M_l\}$ *of segments generated by Algorithm BOE.*

Proof: We first show that the BOE trajectory never crosses essential segments. Assume, for contradiction, that BOE crosses an essential segment E. We show that E is dominated by a necessary segment, a contradiction. The proof depends on the case corresponding to the iteration during which BOE crosses E.

Case 1:

If the extension segment $E(b)$ associated with the obstructing angle b is parallel to E, then $E(b)$ is necessary and dominates E. If $E(b)$ is perpendicular to E, then there must be a side S perpendicular to E that prevents the robot from moving directly towards $E(b)$. In that case, the necessary segment $E(b)$ dominates E, as depicted in Fig. 6.

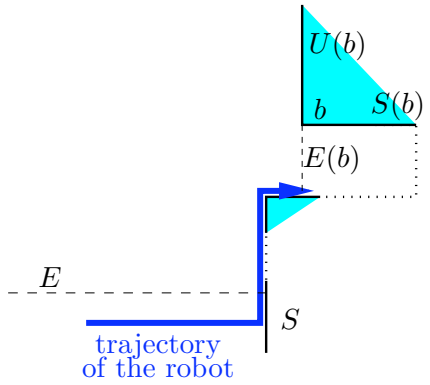

Figure 6: Case 1 with $E(b)$ perpendicular to E

Case 2: Same as Case 1, with f instead of b.

Case 3.1:

If the robot crosses E when moving parallel to the side, then use the proof of Case 3.3, and use the proof of Case 1 otherwise.

Case 3.2:

Same as Case 3.1, with Case 2 instead of Case 1.

Case 3.3:

The robot moves parallel to a side S that is perpendicular to E. The side S' immediately following S in clockwise order forms a $\pi/2$ angle with S, and is at distance strictly greater than one from E. The vicinity segment of S' is clearly necessary and dominates E.

Hence the BOE trajectory does not cross any essential segment.

We prove that each essential segment is either $E(f)$, $E(b)$ or $V(a)$ at the end of some iteration of the loop of Algorithm BOE. By Lemma 3.1 and the fact that the robot has to visit all essential segments to fully explore P, the trajectory BOE visits all the E_i's.

Assume, for contradiction, that BOE visits an essential segment E other than M_i, M_{i+1} or M_{i+2}, on its way from r_i to r_{i+1}. If E is parallel to M_{i+1}, then BOE has to cross E to reach M_{i+1}, since the path from r_i to M_{i+1} is a monotone path, a contradiction.

Now assume that E is perpendicular to M_{i+1}. The path from r_i to M_{i+1} is a monotone path and, since E cannot be crossed, E must intersect M_{i+1} at point r_{i+1}. However, in this case, we have $E = M_{i+2}$. □

We define a *compatible order* of essential segments as follows. In the natural order of essential segments we choose an arbitrary set of disjoint pairs of consecutive intersecting essential segments, and we swap segments in each pair.

Lemma 3.4 *The essential segments are visited in a compatible order $D_1, \ldots, D_m$ by the BOE' trajectory.*

Proof: For $i = 1, \ldots, m$, let S_i be the side associated with the essential segment D_i.

Consider the clockwise order $\mathcal{O}_1$ of sides of the polygon associated with the essential segments. This induces an order $\mathcal{O}_2$ of these essential segments. This order is compatible with the natural order of essential segments. We show that our algorithm visits essential segments in an order compatible with $\mathcal{O}_2$.

The proof is by induction on the S_i's lying in a compatible order between r_0 and f along the boundary of the polygon P. More formally, we show that for all i, if S_j, for $j \geq i+1$, is between S_i and S_{i+1} in the clockwise order along the boundary, then $j = i+2$ and D_{i+1} intersects D_{i+2}.

By Lemma 3.3, each essential segment is in $\{M_1, M_2, \ldots, M_l\}$. If we reach the extension segment D_{i+1} by an iteration corresponding to Cases 2, 3.2 or 3.3, then there is no $j > i+1$ such that S_j is between S_i and S_{i+1} in the clockwise order along the boundary. Indeed, any such side S_j would be entirely in C before visiting D_j, a contradiction.

Assume that D_{i+1} is visited in an iteration corresponding to Cases 1 or 3.1. The boundary between f and b (the obstructing angle) cannot contain any side whose associated segment is essential and parallel to D_{i+1}. Indeed, such a segment would dominate the essential segment D_{i+1}, a contradiction. For the same reason, there can be no sides whose associated segment is essential and perpendicular to D_{i+1} without intersecting it. So, there can only be essential segments perpendicular to D_{i+1} and intersecting it. There can be only one such segment, and it must be D_{i+2} because an essential segment cannot be dominated. When reaching D_{i+2}, the number $j = i+2$ is the only integer greater than $i+1$, such that the side S_j is between S_i and S_{i+1} in clockwise order. □

In order to compare the BOE trajectory to the GE trajectory, we define a trajectory GC that greedily visits essential segments in the same compatible order as BOE. For $i \in 1, \ldots, l$, we denote by y_i the point on D_i at the minimum distance from point y_{i-1}, with $r_0 = y_0 = y_{m+1}$. More formally, the trajectory GC is the one following a shortest path from y_{i-1} to y_i, for all $1 \leq i \leq m+1$.

Lemma 3.5 *The GC trajectory has the same length as GE.*

Proof: Let δ denote the permutation on $\{1, 2, \ldots, m\}$ such that $E_{\delta(i)} = D_i$. By definition of GC and of a compatible order, for each $i = 1, \ldots, m$, one of the following holds:

$\delta(i) = i$ or

$\delta(i) = i + 1$ and $E_i = D_{i+1}$ intersects $E_{i+1} = D_i$.

By induction on i we can show (cf. [5]) that either

$|\,GE[r_0, x_i]| = |\,GC[r_0, y_i]|$ and $x_i = y_i$ (case $\delta(i) = i$), or

$|\,GE[r_0, x_{i+1}]| = |\,GC[r_0, y_{i+1}]|$ and $x_{i+1} = y_{i+1}$ (case $\delta(i) = i + 1$).

Applying the above for $i = m$ proves the lemma. □

Now, we are ready to state the key lemma for the proof of Theorem 3.1.

Lemma 3.6 *The BOE' trajectory has the same length as the GC trajectory.*

Proof: By Lemma 3.3 and Lemma 3.4, we can define an increasing function γ: $\gamma(j) = i$ if $D_j = M_i$. Thus, $D_j = M_{\gamma(j)}$. For convenience, we also define its "inverse" $\omega : \omega(i) = j$, if and only if, $\gamma(j-1) < i \leq \gamma(j)$. Thus, when the robot is at z_i , the last essential extension visited was $D_{\omega(i)-1} = M_{\gamma(\omega(i)-1)}$. Here, we define $D_0 = M_0$ to be the side containing the starting point z_0. We prove by induction on i, that one of the following conditions holds.

H1. BOE' reaches M_i at z_i, so that its trajectory from $y_{\omega(i)-1}$ (a point on the last visited essential segment $D_{\omega(i)-1}$) to z_i is a shortest path from $y_{\omega(i)-1}$ to M_i.

H2. Let i' be the highest index in the range $\gamma(\omega(i) - 1) \leq i' \leq i - 1$, such that $M_{i'}$ and M_i intersect. BOE' reaches M_i at the intersection $X(M_i, M_{i'})$ of M_i and $M_{i'}$, via a shortest path from $y_{\omega(i)-1}$ to $X(M_i, M_{i'})$.

Starting at $z_0 = y_0$, let $D_0 = M_0$ be the side of P containing z_0. The claim (in this case, H1) is trivially true for $i = 1$. In general, suppose the hypothesis is true when the robot is at point z_i, and consider the next point z_{i+1}. By definition of BOE', we reach the next segment M_{i+1} from z_i by a shortest path.

Case 1: H1 is true for i.

If M_i is an essential segment, then $M_i = D_{\omega(i+1)-1}$ and $z_i = y_{\omega(i+1)-1}$. By definition of BOE', we reach M_{i+1} via a shortest path from z_i. This proves H1 for $i + 1$.

On the other hand, if M_i is not an essential segment, then $\omega(i + 1) = \omega(i)$. Thus, the last essential segment visited from z_{i+1} is $D_{\omega(i)-1}$, visited at point $y_{\omega(i)-1}$.

If M_{i+1} does not intersect M_i, then M_{i+1} dominates M_i according to $y_{\omega(i)-1}$. We reach M_{i+1} at z_{i+1} from $y_{\omega(i)-1}$ via a shortest path from $z_{\gamma(\omega(i)-1)} = y_{\omega(i)-1}$ to M_i, reaching it at z_i (induction hypothesis), followed by a shortest path from z_i to M_{i+1} (by definition of BOE'). Since M_{i+1} dominates M_i according to $y_{\omega(i)-1}$, this is a shortest path from $y_{\omega(i)-1}$ to M_{i+1} by Proposition 2.2. This proves H1 for $i + 1$.

If M_{i+1} intersects M_i, then BOE' reaches M_{i+1} (moving along M_i) at the intersection point $X(M_i, M_{i+1})$. Since the shortest path from $z_{\gamma(\omega(i)-1)} = y_{\omega(i)-1}$ to M_i meets M_i at z_i by the induction hypothesis, this path followed by the line segment from z_i to $X(M_i, M_{i+1})$ is a shortest path from $y_{\omega(i)-1}$ to $X(M_i, M_{i+1})$, by Proposition 2.1. Thus, H2 holds for $i + 1$.

Case 2: H2 is true for i.

If M_i is an essential segment, then $M_i = D_{\omega(i)}$ and $z_i = y_{\omega(i)}$. By definition of BOE', we reach M_{i+1} via a shortest path from z_i. This proves H_1 for $i+1$.

If M_i is not an essential segment, then $\omega(i+1) = \omega(i)$. Thus, the last essential segment visited from z_{i+1} is $D_{\omega(i)-1}$, visited at point $y_{\omega(i)-1}$.

Suppose that M_{i+1} dominates M_i according to y_{i-1}, hence also according to $y_{\omega(i)-1}$, and that M_{i+1} also dominates $M_{i'}$ according to $y_{\omega(i)-1}$. Then z_{i+1} and $y_{\omega(i)-1}$ are in opposite quadrants defined by M_i and $M_{i'}$. By Proposition 2.3, there is a shortest path from $y_{\omega(i)-1}$ to M_{i+1} passing through the intersecting point $X(M_i, M_{i'})$. Due to the fact that the path from $z_i = X(M_i, M_{i'})$ to z_{i+1} is a shortest path, H1 holds for $i+1$.

If M_{i+1} does not dominate both M_i and $M_{i'}$ according to $y_{\omega(i)-1}$, it must intersect either M_i or $M_{i'}$. Hence, either M_{i+1} intersects M_i, dominating $M_{i'}$ according to $y_{\omega(i)-1}$ or M_{i+1} intersects $M_{i'}$, dominating M_i according to $y_{\omega(i)-1}$. In both cases, we move along the segment $M_{i''}$ ($i'' = i$ or $i'' = i'$) intersecting M_{i+1} to reach the intersection point $z_{i+1} = X(M_{i+1}, M_{i''})$. Applying Proposition 2.3 with $M = M_{i+1}$, $M' = M_{i''}$, $u = y_{\omega(i)-1}$ and $v = X(M_{i+1}, M_{i''})$, we obtain that the path from $u = y_{\omega(i)-1}$ to $X(M_{i+1}, M_{i''})$ is a shortest path. Hence H2 holds for $i+1$.

Therefore, in either case, H1 or H2 holds for $i+1$, and hence one of these conditions holds for all i, $1 \leq i \leq k$, by induction.

If H1 holds for $D_j = M_i$ then the trajectory BOE' follows a shortest path from y_{j-1} to D_j. On the other hand, if H2 holds then the trajectory BOE' follows a shortest path from y_{j-1} to $X(M_i, M_{i'})$ with $D_j = M_i$. Assume that $M_{i'}$ is necessary. Note that $M_{i'}$ is the last encountered segment of all the segments perpendicular to M_i, so no segment parallel to $M_{i'}$ can dominate $M_{i'}$. On the other hand, any extension or vicinity segment perpendicular to $M_{i'}$ and dominating $M_{i'}$ would be necessary and would dominate M_i, a contradiction. So, $M_{i'}$ is essential and $M_{i'} = D_{j-1}$. Hence, BOE' follows a shortest path from y_{j-2} to D_j that visits D_{j-1}.

Now, if we assume that $M_{i'}$ is not necessary, then when the robot meets $M_{i'}$, the entire boundary of the foreign polygon $FP(M_{i'})$ is in C, since f and z are in the other subpolygon of $M_{i'}$ when BOE' visits $M_{i'}$. Since the side S_{j+1} associated with D_{j+1} is not in C when we meet $M_{i'}$, it follows that either D_{j+1} intersects $M_{i'}$ or dominates $M_{i'}$ according to y_{j-1}. The first case cannot happen, since D_{j+1} would be parallel to D_j and dominate D_j, a contradiction. In the second case, the path from y_{j-1} to D_{j+1} is a shortest path visiting D_j, by Proposition 2.4.

We can now conclude that, for all $1 \leq j \leq l$, BOE' follows either a shortest path from y_{j-1} to D_j or a shortest path from y_{j-1} to D_{j+1} that visits D_j. This proves the lemma by the definition of GC. □

Proof of Theorem 3.1: Any boundary exploration trajectory (including the optimal one) has length not smaller than that of GE, by Lemma 2.2. By Lemma 3.5, we have $|GE| = |GC|$. By Lemma 3.6, we have $|BOE'| = |GC|$. By Lemma 3.2, we have $|BOE'| = |BOE|$. By Lemma 3.1, BOE is a boundary exploration trajectory. Hence BOE is an optimal boundary exploration trajectory. □

3.2 Negative results

In this section we show that in all scenarios except the one covered by Theorem 3.1, optimal on-line exploration is impossible.

Lemma 3.7 *There is no optimal on-line algorithm for the exploration of rectilinear polygons, with a square SA, in the L_1-metric, even with the starting point at the boundary.*

Proof: We consider two polygons W and T depicted in Fig. 7, and the exploration problem starting from the point x at the boundary of each of these polygons.

Notice that the visible parts of the two polygons are identical when the robot is at any point inside the rectangle $abkl$, the boundary of the rectangle included. So, the adversary can arbitrarily choose one of the two polygons when the robot leaves this rectangle to explore the rest of the polygon. The adversarial strategy to prevent optimality consists in taking the polygon T, if the robot exits the rectangle $abkl$ through point k or b, and in taking the polygon W otherwise.

We first show that any exploration trajectory passing through point b or k in polygon T is not optimal. Note that the order in which the two angles f and g are explored does not matter because of the symmetry of polygon T. The exploration trajectory R of T depicted in Fig. 7 is optimal, since the trajectory follows shortest paths to explore the angle f (at point y) and then the angle g (at point z) starting from point x.

Let us assume, for contradiction, that there is an optimal exploration trajectory E passing through b. In order to have the same length as R, the trajectory E must follow shortest paths from x to b, from b to y, from y to z and from z to x. Let us consider the square region Q of points at distance at least one from lines ab and bk, and at distance at least two from lines lk and gf. The interior points in Q and the points of side kl cannot be explored by a robot following a shortest path xb, by or xy, since these points are at distance larger than one from any shortest path connecting these pairs of points. Consequently, the points of Q and those in the side kl need to be explored when moving on the trajectory between z and x. To explore the points of Q, the robot has to move past the line kl and continue moving east for a distance strictly greater than one. From the fact that this must be a shortest path to x, the robot cannot move west after this move and so cannot explore points of the side kl. Hence, the trajectory E is not an exploration trajectory and so there is no optimal exploration trajectory passing through b. By symmetry of the polygon, the same is true for point k.

We now show that any optimal exploration trajectory passing through any point t of the segment bk (ends of the segment excluded) in polygon W exits the rectangle $abkl$ through b or k. Note that any optimal trajectory needs to explore the angles e or h before the angles f or g. Indeed, there is a shortest path from the point t to a point from which f is visible (respectively the angle g) that explores the angle e (respectively the angle h). Consequently, any optimal exploration trajectory needs to follow an optimal path from t to explore one of the two angles e or h. These paths exit the rectangle $abkl$ through point b or k, and so no optimal exploration trajectory can exit this rectangle through an inside point of the segment.

□

Lemma 3.8 *There is neither an optimal on-line algorithm for the exploration, nor for the boundary exploration of rectilinear polygons with a square SA, in the L_1-metric, starting at an arbitrary point of the polygon.*

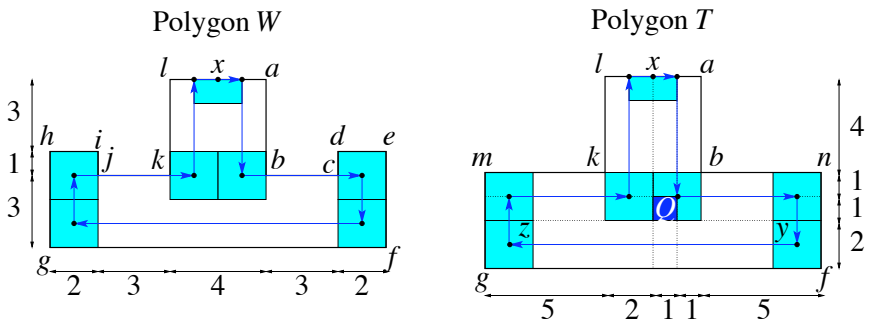

Figure 7: Optimal solutions in polygon W and T

Proof: The idea behind the proof is that the robot cannot figure out the shortest path to see the closest point of the boundary. To prevent exploration or boundary exploration optimality, the adversary takes a square $abcd$ of side length 4 and places the robot so that the nearest side of the square (without loss of generality the side ad) is in the opposite direction of the first segment of the robot's trajectory, at a distance slightly larger than 1 from this side, as depicted in Fig. 8.

Notice that, since we use the L_1-metric, there is a shortest path between any pair of points from which opposite angles of the square are visible, that visits one of the other angles. For instance, a shortest path, from a point where a is seen to a point where c is seen, moves parallel to side ab and then moves parallel to side bc, exploring the angle b. So, any optimal trajectory exploring the four angles explores these angles in cyclic order. The optimal trajectory to explore the four angles consists in moving toward one of the two closest angles (a or d), exploring other angles cyclicly using shortest paths and coming back to the starting point. Let B be the boundary of the square inside $abcd$ with sides at distance one from those of $abcd$. Using a shortest path from the starting point to B, then going around B and getting back to the starting point, we obtain an optimal exploration trajectory. Any trajectory that first moves away from B is strictly longer, and so is not optimal.

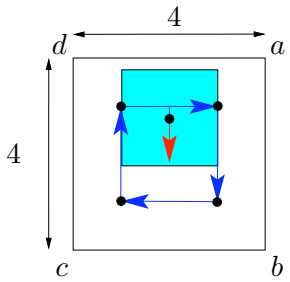

Figure 8: Optimal trajectory for exploration starting from an inside point of the polygon

□

Lemma 3.9 *There is neither an optimal on-line algorithm for the exploration, nor for the boundary*

exploration of rectilinear polygons with a round SA, in the L_1-metric, even with the starting point at the boundary.

Proof: We consider two polygons, a rectangle $abcd$ of length $l > 3$ and width 2 and a L-shaped polygon $abcefd$ of the same length and width (cf. Fig. 9). Note that for the two polygons, any boundary exploration trajectory is an exploration trajectory since all points inside these polygons are at distance at most one from the boundary. Let a be the starting point in both polygons.

For the L-shaped polygon, the optimal exploration trajectory starting from the angle a is the rectangle $ageh$ of length $l-1$ and width 1. Indeed, this trajectory clearly minimizes the x-axis and y-axis shifting of the robot to explore the angles b and d. Moreover, this trajectory is unique if we disregard the orientation, since any optimal exploration trajectory has to pass through g, e and h exploring angles b, c and d at distance at least one of the lines bc and fd.

The adversarial strategy consists in taking the L-shaped polygon, if the robot does not initially follow the side ab, or moves nearer than distance one from the side bc, and in using the rectangle otherwise. Notice that if the robot initially follows a side, then we can assume that it is the side ab, since the adversary can rotate the figure, if the robot follows the other side.

In the first case, the exploration trajectory is non-optimal since it differs from the unique optimal trajectory for the L-shaped polygon. In the second case, the shortest exploration trajectory for the rectangle, that does not move nearer than distance one of side bc, is the rectangle $aged$ of length l and width 1, since the robot can only explore the angle c from the point e. This trajectory is strictly longer than some exploration trajectories without the constraint, such as the rectangle of length $l-\sqrt{2}/2$ and width $2-\sqrt{2}/2$ depicted in Fig. 9. Note that the robot cannot decide in which of the two polygons it is, before seeing point e, since the adversary can freely adjust the length l.

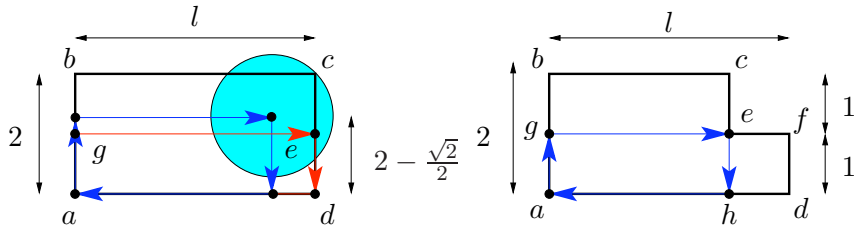

Figure 9: Optimal trajectories in rectangle and the L-shaped polygon

□

Lemma 3.10 *There is neither an optimal on-line algorithm for the exploration, nor for the boundary exploration of rectilinear polygons with a square or a round SA, in the L_2-metric, even with the starting point at the boundary.*

Proof: We consider two squares, a small square $abcd$, that is inscribed in the SA of the robot (a square of side length $\sqrt{2}$ for the round SA and of side length 2 for the square SA) and a large square $a'b'c'd'$ (square of side length 2 for the round SA and of side length 3 for the square SA), as depicted in Fig. 10. We consider the boundary exploration problem in the L_2-metric, with the starting point $a = a'$ in both squares. Observe that from this starting point the view of the robot is exactly the same in both squares.

For the small square, the optimal trajectory consists in moving toward the diagonal angle c from a until exploring it, and coming back to a following the same way. Indeed, the robot will explore the entire square when reaching its center, and this is the unique optimal boundary exploration trajectory, since it is the only shortest path to explore the angle c. The angle α formed between side ab and the direction followed by the robot is $\pi/4$.

Let us assume that there is an optimal boundary exploration trajectory for the large square exploring angles in a non cyclic order. The trajectory intersects itself, since the robot must visit diagonal angles consecutively. By Proposition 2.5, we can obtain a boundary exploration trajectory using the same parts of the trajectory but without crossings. So, the robot must visit the angles in a cyclic order. To explore the first angle b' (or d') optimally, the robot must move straight to the set of points from which b' can be seen. The angle β formed by this direction and by the side $a'b'$ (or $a'd'$) is strictly less than $\pi/4$, since by moving straight at angle $\pi/4$ the robot can never explore b' or d'.

Let us describe the adversarial strategy against the robot to prevent optimality of boundary exploration. If the robot begins its trajectory in a direction at angle different than $\pi/4$ from the side, then the adversary takes the small square, otherwise it takes the large square. In the two cases, the robot's trajectory is clearly suboptimal, since it differs from any optimal trajectory.

This example holds for the exploration problem as well, since any optimal boundary exploration trajectory is an exploration trajectory in both squares. This is trivially true for the small square. For the large square in the case of round SA, any point inside the square is at distance at most one from the boundary, and so must be explored by a robot that follows a boundary exploration trajectory. For the large square in the case of square SA, it suffices to remark that any optimal trajectory cannot be at distance less than one from sides $b'c'$ and $c'd'$, and that any point inside the square is either at distance less than one from the boundary part $b'a'd'$, or at distance less than two from the boundary part $b'c'd'$.

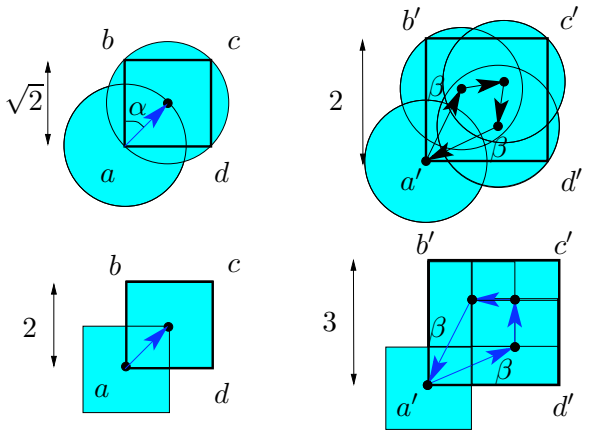

Figure 10: Optimal solutions in the small and the large square

□

Theorem 3.1 and Lemmas 3.7, 3.8, 3.9, 3.10 imply the following result that completely solves the optimality problem of on-line exploration of rectilinear polygons (see Table 1).

Problem	Starting point in	SA shape	Metric	Optimality
boundary exploration	boundary	square	L_1	optimal by Theorem 3.1
exploration	boundary	square	L_1	non-optimal by Lemma 3.7
both	interior	square	L_1	non-optimal by Lemma 3.8
both	both	round	L_1	non-optimal by Lemma 3.9
both	both	both	L_2	non-optimal by Lemma 3.10

Table 1: Solution of the optimality of on-line exploration

Theorem 3.2 *The only case where on-line exploration of rectilinear polygons can be optimal is the case of the boundary exploration with square SA in the L_1-metric, starting at the boundary. Algorithm BOE performs optimal boundary exploration in this case.*

4 Competitiveness

4.1 Rectilinear polygons and square sensing area

Theorem 4.1 *There exists a competitive on-line algorithm for exploration and for boundary exploration of rectilinear polygons with square SA for both metrics and for any starting point.*

Proof: Note that we can restrict attention to the case of the L_1-metric, since any competitive on-line algorithm in the L_1-metric is competitive in the L_2-metric [5]. First perform a boundary exploration of the polygon. At this point the entire polygon is known to the robot. Then use the off-line polynomial exploration algorithm that gives a 4/3-approximation of the optimal [15]. Hence, any competitive on-line algorithm for boundary exploration gives a competitive algorithm for exploration, and we can restrict attention to boundary exploration.

If the starting point s is on the boundary, then use Algorithm BOE. Otherwise, consider a variant of BOE, called BOE^*, in which the robot starts and ends at a point s inside the polygon and sees a non-empty part of the boundary. Choose any continuous part C of the boundary seen by the robot located at s and let r_0 be the first end of C in clockwise order. Algorithm BOE^* follows the same rules as BOE. Note that in BOE the fact of starting at the boundary was not used to describe the algorithm, but only to prove the optimality of the boundary exploration.

Case A: A part of the boundary is seen by the robot at the starting point s.

Use algorithm BOE^*.

Case B: No part of the boundary is explored by the robot at s.

Choose a direction (half-line starting at s) and move along this direction until Case A occurs. Then, apply strategy for Case A and return to s by a shortest path.

Consider Case A. We define essential segments according to s and we set the first segment encountered in clockwise order from r_0 to be E_1 and the other segments $E_2, \ldots, E_m$ are ordered

in clockwise order starting from E_1. Similarly as in the proof of Theorem 3.1, we have $|GE| = |BOE^*|$ and this length is equal to the distance traveled by the robot in Case A. Let GE_i be the trajectory that greedily visits segments in the order $E_i, E_{i+1}, \ldots, E_m, E_1, \ldots, E_{i-1}$. By Lemma 2.1, there exists a j such that GE_j is a shortest trajectory that visits all the E_i's starting and ending at s, since greedily visiting essential segments in clockwise order is optimal. Any boundary exploration trajectory starting and ending at s is no shorter than GE_j, since such a trajectory has to visit all the essential segments. For all i, we have $|GE| \leq 2|GE_i|$ [5]. Hence, the distance traveled by the robot in Case A is at most twice the length of any optimal trajectory for boundary exploration. The distance traveled by the robot in Case B is at most twice the length of an optimal trajectory for boundary exploration plus the length of the trajectory in Case A. Hence the algorithm is competitive in both cases. □

4.2 Arbitrary convex polygons

In this section we present a competitive exploration algorithm, called Algorithm `Convex`, working for arbitrary convex polygons, for round or square SA and regardless of the starting point. We use the L_2-metric, and the result holds for the L_1-metric as well by changing the competitive constant.

Before explaining the idea of our algorithm, let us remark that the naive exploration method (choose any direction, go to the boundary, trace it and then use the best off-line algorithm for the already known polygon) is not a competitive algorithm. Indeed, consider a starting point in the center of a square of side $2 + \epsilon$, for an arbitrarily small $\epsilon > 0$. The cost of the optimal algorithm is less than 5ϵ, while the naive approach costs more than 1.

The idea of Algorithm `Convex` is the following. First move along a direction until a boundary point becomes explored. Call this distance δ. This is safe, as the optimal algorithm must travel at least the distance $\delta/\sqrt{2}$. Then move along boundaries of increasing squares centered at the starting point, of sizes 2δ, 4δ, 8δ, and so on, until the entire polygon is explored, or until the size of the square is at least 1. (If the boundary of the polygon to be explored prevents the robot from continuing on the square, then it "slides" on the boundary, returning to the travel on the square when again possible.) Since sizes of squares are doubled at each stage, the total trajectory length is at most the double of traversing the last square. If the whole polygon has been already explored, then the trajectory length is proportional to that of the optimal algorithm. Otherwise, the optimal trajectory length is proportional to the diameter and both these values must be at least 1/4. The trajectory length up to this moment is constant, hence making the tour of the polygon boundary and then applying the optimal off-line algorithm to explore its interior (at this point the polygon is known) is proportional to the diameter and hence competitive.

Algorithm `Convex`

Phase 1. Let p be the starting point of the robot. If the entire polygon P is included in the SA, then stop. Otherwise, choose a direction (half-line starting at p) α on which no boundary point is in the SA and move along α until a boundary point on α is in the SA. Denote by q the position of the robot at this moment. Let δ be the distance between p and q.
Phase 2. Define the family $S_1, S_2, ...$ of squares centered at p, with sides parallel and perpendicular to α and such that S_i has side of length $2^i\delta$. For each i, let T_i be the boundary of the polygon resulting from the intersection of square S_i with polygon P. Phase 2 is divided into stages $1, 2, \ldots$. The aim of stage i is executing the tour of T_i. Each stage i ends at a point q_i of the half-line α. Stage 1 starts and ends at point $q_1 = q$. Recursively, if q_{i-1} is at the boundary of the polygon to be explored then $q_i = q_{i-1}$. If q_{i-1} is in the interior of the polygon to be explored then, in stage i, the robot moves along α from q_{i-1}, away from p, to the point q_i, intersection of α and T_i. (This point may be on the boundary of the polygon, or on the boundary of the square S_i.) Then the robot makes a complete tour of T_i, ending in q_i. The last stage of Phase 2 is when the entire polygon P is explored, or when $2^i\delta \geq 1$, whichever comes first.
Phase 3. If Phase 2 ended because the entire polygon P was explored, then Phases 3 and 4 are void. Otherwise, the robot moves from point q_i at which Phase 2 ended, along the half-line α, away from p, to the closest point at the boundary of the polygon P, and makes a complete clockwise tour of this boundary. Denote by r the point of the boundary at which Phase 3 ends.
Phase 4. At the end of Phase 3 the robot knows the polygon, although it may have not explored all its interior points yet. If it has, then stop. Otherwise, the robot goes back to p along the half-line α and applies the optimal exploration algorithm for the polygon, starting from point p; then it stops.

Theorem 4.2 *Algorithm* `Convex` *is a competitive algorithm to explore any convex polygon, starting from any point, for round or square SA, and for the L_1 or the L_2-metric.*

Proof: Let D denote the diameter of the polygon to be explored, B the length of its boundary, OPT the length of the optimal exploration trajectory of a robot knowing the polygon and starting at point p, and L the total length of the trajectory of the robot starting at point p and using Algorithm `Convex`. For each point x in P, let $g(x)$ be the minimum distance from p to a point from which x is visible by the robot. Let $G = \max_{x \in P}\{g(x)\}$. Notice that $G \leq OPT$. Observe as well that $G \geq \delta$ for a round SA and $G \geq \delta/\sqrt{2}$ for a square SA. Indeed, the second inequality is justified as follows. Since α intersects the boundary of SA at angle at least $\pi/2$, the distance from SA to the farthest point of P on the half-line α is at least $\delta/\sqrt{2}$.

Consider Phase 2 of the algorithm and suppose that it lasted i stages. If Phase 2 ended and P has not yet been explored, then $2^i\delta \geq 1$. At this point the robot has performed a tour of T_{i-1}. Since the polygon is convex, all the points visible from points inside T_{i-1} have been explored. In this case $G \geq 1/4$, since every point that can be explored from a point at distance $1/4$ from p was explored. Notice that the length of the boundary of T_i is at most $4 \cdot 2^i\delta$, for each i, in view of the convexity of P. The length of the trajectory of our robot in Phase 1 is at most $2\sqrt{2} \cdot OPT$, in Phase 2 it is $O(1) + 5\sqrt{2} \cdot OPT$, in Phase 3 it is $O(1) + B$, and in Phase 4 it is $O(1) + OPT$. Hence there exists a positive constant c_1 such that $L \leq c_1 + B + 15 \cdot OPT$. Since the polygon is convex, we have $B \leq \pi D$. Notice that $D \leq 2G + 2\sqrt{2}$ since the distance from a point x to to the point p is at most $g(x) + \sqrt{2}$. Hence, we have $D \leq 14G$, since $G \geq 1/4$. It follows that $B \leq 14\pi G$. As a

consequence, we get:

$$\begin{aligned} L &\leq c_1 + B + 15 \cdot OPT \\ &\leq c_1 + 14\pi G + 15 \cdot OPT \\ &\leq c_1 + (14\pi + 15)OPT \end{aligned}$$

Since $OPT \geq G \geq 1/4$, we have $L \leq c_2 \cdot OPT$, for some positive constant c_2.

Hence we may assume that the end of Phase 2 was caused by exploring the entire polygon P. If this phase ends after stage 1, then $L \leq 9\delta$. Hence, $L \leq 9\sqrt{2} \cdot OPT$, since $OPT \geq \delta/\sqrt{2}$.

Thus we may suppose that Phase 2 lasted $i \geq 2$ stages. Let $x = 2^i\delta$. Since the polygon is convex, all the points visible from points inside T_{i-1} have been explored. Since at the end of stage $i-1$ the entire polygon has not yet been explored, it follows that $G \geq x/4$. The length of the trajectory of our robot until the end of Phase 2 is at most $10x$. Hence $L \leq 40G \leq 40 \cdot OPT$.

It follows that $L \leq c \cdot OPT$ holds in all cases for some positive constant c, and hence Algorithm `Convex` is competitive. □

5 Conclusion

For the problem of optimality of on-line exploration of rectilinear polygons, our results explain the situation in each of the considered scenarios: we gave an optimal boundary exploration algorithm for a robot with square sensing area starting at the boundary, in the Manhattan metric, while in all other scenarios (exploration of the entire polygon, or arbitrary starting point, or round SA, or the Euclidean metric) we proved that optimal on-line exploration is impossible.

For the problem of competitiveness of on-line exploration of rectilinear polygons, our results are less complete: we showed a competitive algorithm for a robot with square sensing area, regardless of the metric and of the starting point. It is natural to ask if the same result is true for a round sensing area. We conjecture that the answer to this question is positive. It should be noted that competitiveness for the round SA does not immediately follow from competitiveness for the square SA, because there is no bound on the ratio between the lengths of optimal exploration trajectories in these scenarios.

An even bigger challenge would be to show a competitive on-line exploration algorithm for arbitrary polygons, for both shapes of the sensing area. Our competitive algorithm for convex polygons is a step in this direction.

References

[1] S. Albers, K. Kursawe, and S. Schuierer. Exploring unknown environments with obstacles. *Algorithmica*, 32(1):123–143, 2002.

[2] E. M. Arkin, S. P. Fekete, and J. S. B. Mitchell. Approximation algorithms for lawn mowing and milling. *Comput. Geom. Theory Appl.*, 17(1-2):25–50, 2000.

[3] W. Chin and S. Ntafos. Optimum watchman routes. In *Proc. of Symposium on Computational Geometry (SCG 1986)*, pages 24–33, 1986.

[4] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment. In *Proc. of Foundations of Computer Science (FOCS 1991)*, pages 298–303, 1991.

[5] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment: the rectilinear case. *J. ACM*, 45(2):215–245, 1998.

[6] Y. Gabriely and E. Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. In *Int. Conf. of Robotics and Automaton (ICRA 2001)*, volume 2, pages 1927–1933, 2001.

[7] S. Ghosh, J. Burdick, A. Bhattacharya, and S. Sarkar. Online algorithms with discrete visibility - exploring unknown polygonal environments. *Robotics & Automation Magazine*, 15(2):67–76, 2008.

[8] M. Hammar, B. J. Nilsson, and S. Schuierer. Improved exploration of rectilinear polygons. *Nordic J. of Computing*, 9(1):32–53, 2002.

[9] M. Held. *On the computational geometry of pocket machining.* Springer-Verlag New York, Inc., 1991.

[10] F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. The polygon exploration problem. *SIAM J. Comput.*, 31(2):577–600, 2001.

[11] C. Icking, T. Kamphans, R. Klein, and E. Langetepe. On the competitive complexity of navigation tasks. In *Revised Papers from the International Workshop on Sensor Based Intelligent Robots*, pages 245–258, London, UK, 2002. Springer-Verlag.

[12] C. Icking, T. Kamphans, R. Klein, and E. Langetepe. Exploring simple grid polygons. In *In 11th Internat. Comput. Combin. Conf*, pages 524–533, 2005.

[13] J. M. Kleinberg. On-line search in a simple polygon. In *Proc . ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 8–15, 1994.

[14] B. M. E. Moret, M. Collins, J. Saia, and L. Yu. The ice rink problem. In *Proc. of the 1st Workshop on Algorithm Engineering*, pages 104–111, 1997.

[15] S. Ntafos. Watchman routes under limited visibility. *Comput. Geom. Theory Appl.*, 1(3):149–170, 1992.