Competitive Searching in a Generalized Street

Amitava Datta [2]

The University of Western Australia, Department of Computer Science, Perth, WA 6907, Australia

Christian Icking [3,1]

FernUniversität Hagen, Praktische Informatik VI, 58084 Hagen, Germany

Abstract

We consider the problem of a robot which has to find a target in an unknown simple polygon, based only on what it has seen so far. A *street* is a polygon for which the two boundary chains from start to target are mutually weakly visible. A target inside a street can be found by walking a path that is at most a constant times longer than the shortest path in the street from start to target. We define a strictly larger class of polygons, called *generalized streets* or $\mathcal{G}$-streets, which are characterized by the property that every point on the boundary of a $\mathcal{G}$-street is visible from a point on a horizontal line segment connecting the two boundary chains. We present an on-line strategy for a robot to find the target in an unknown rectilinear $\mathcal{G}$-street; the length of its path is at most 9 times the length of the shortest path in the L_1 metric, and 9.06 times the length of the L_2-shortest path. These bounds are optimal.

Key words: Simple polygon, street, searching, doubling, competitive.

1 Introduction

Path planning is one of the fundamental problems in robotics. In contrast to path planning with complete information, i. e. if the whole scene is known in

[1] Corresponding author, email icking@feu.de

[2] This work was done when the author was visiting FernUniversität Hagen, and the author would like to thank Rolf Klein for his support.

[3] Supported by the Deutsche Forschungsgemeinschaft, grant Kl 655/8-3.

advance, it is interesting for many real life situations to consider the problem of finding a path on-line. Here, the geometry of the scene is unknown and the robot has to make decisions depending on the information it gathers through (e. g. visual or tactile) sensors.

Lumelsky and Stepanov [30] study this problem when only tactile sensors are used. Papadimitriou and Yannakakis [31] consider several variants of the problem as a two-person game and give bounds on the length of the generated path in terms of the length of the shortest path. Blum et al. [3] present deterministic and randomized on-line algorithms for several versions of this problem when the obstacles are rectangles or convex polygons.

Following the concept of *competitive algorithms* introduced by Sleator and Tarjan [34] for general problems in computer science, a strategy for searching on-line is called *k-competitive* for a constant k, if it can be guaranteed that the generated path is no longer than k times the shortest path. The constant k is then called the *competitive factor*.

Klein [20] describes the problem of designing an on-line strategy for a robot when the robot is constrained to move inside a simple polygon from a vertex s to a vertex t. He gives an on-line strategy for a class of polygons called *streets*. A street is a polygon such that the clockwise polygonal chain L from s to t and the anticlockwise polygonal chain R from s to t are mutually weakly visible. An alternative criterion for recognizing a street is also given by Icking and Klein [13]. Das et al. [4] and Tseng et al. [35] give algorithms to compute, for a given polygon, all pairs of points s and t such that the polygon is a street with respect to these points.

Klein proves a lower bound of $\sqrt{2}$ (≈ 1.41) on the competitive factor for searching in a street, and his strategy achieves a competitive factor of $1 + \frac{3}{2}\pi$ (≈ 5.72). Kleinberg [21] improves this ratio to $2\sqrt{2}$ (≈ 2.83). He also mentions that his strategy is $\sqrt{2}$-competitive for rectilinear streets which is optimal. Interestingly, this means in fact that one can find on-line an L_1-shortest path for rectilinear streets. Quite a couple of improvements have been made to the competitive factor for searching in streets, e. g. by López-Ortiz and Schuierer [24,25,27] and by Icking et al. [17], to name just a few. This question is now finally settled by an optimal $\sqrt{2}$-competitive strategy by Icking et al. [16] and, independently, by Schuierer et al. [33].

Other interesting geometric applications for competitive strategies are e. g. the search for the kernel of a polygon [15,24,28,22], exploration or path planning in unknown environments [7,11,12,1,6], or localization in known environments [8,10,14,19,32].

It has been posed as a challenging open problem whether more general classes

of polygons admit competitive searching. In this paper, we introduce such a class of polygons, namely the *generalized streets*, or $\mathcal{G}$-streets for short. We give a strategy for searching in the *rectilinear* case which is 9-competitive in comparison to the L_1-shortest path and 9.06-competitive for L_2. Both bounds are optimal, the lower bound of 9.06 has been found by López-Ortiz and Schuierer [26], who also describe an 80-competitive strategy for searching in the non-rectilinear case. Related results exist about competitive searching in star-shaped polygons [23,24,28], competitiveness with respect to the link distance [9] and further generalized polygons [5].

The rest of the paper is organized as follows. In Section 2, we introduce some definitions and properties related to $\mathcal{G}$-streets. We give the strategy for rectilinear $\mathcal{G}$-streets and analyze its complexity in Section 3. Some further results and open questions are mentioned in Section 4.

2 Definitions and properties

We consider a simple polygon P in the plane with a start vertex, s, and a target vertex, t. We use the Cartesian coordinate system. The notion of *horizontal* (resp. *vertical*) indicates a direction parallel to the x-axis (resp. y-axis).

For simplicity of the presentation, we assume general position, i. e. no three vertices are aligned, which implies that no two non-consecutive vertices are horizontally aligned. Our results can be easily adapted to handle the general case when three vertices are aligned or s and t are not vertices. The clockwise (resp. anticlockwise) chain from s to t is called the *left chain* or L (resp. *right chain* or R). These two chains induce a natural ordering of vertices of the polygon from s to t.

Definition 1 **[20]** A polygon is called a *street* if the two chains L and R are mutually weakly visible.

Definition 2 A horizontal line segment inside the polygon P with both endpoints on the boundary of P is called a *chord*. The extension of a horizontal edge e across the interior of P such that the two end points hit the polygon boundary is called the *chord of* e, analogously we define the *chord of a vertex*. If a chord touches both the L and R chains it is called an *LR-chord*.

Definition 3 A simple polygon in the plane is called a *generalized street* or *$\mathcal{G}$-street* if for every boundary point $p \in L \cup R$, there exists an LR-chord c such that p is visible from a point on c.

Figure 1 shows an example of a $\mathcal{G}$-street. Some LR-chords, drawn as dashed

lines, indicate visibilities along the dotted rays.

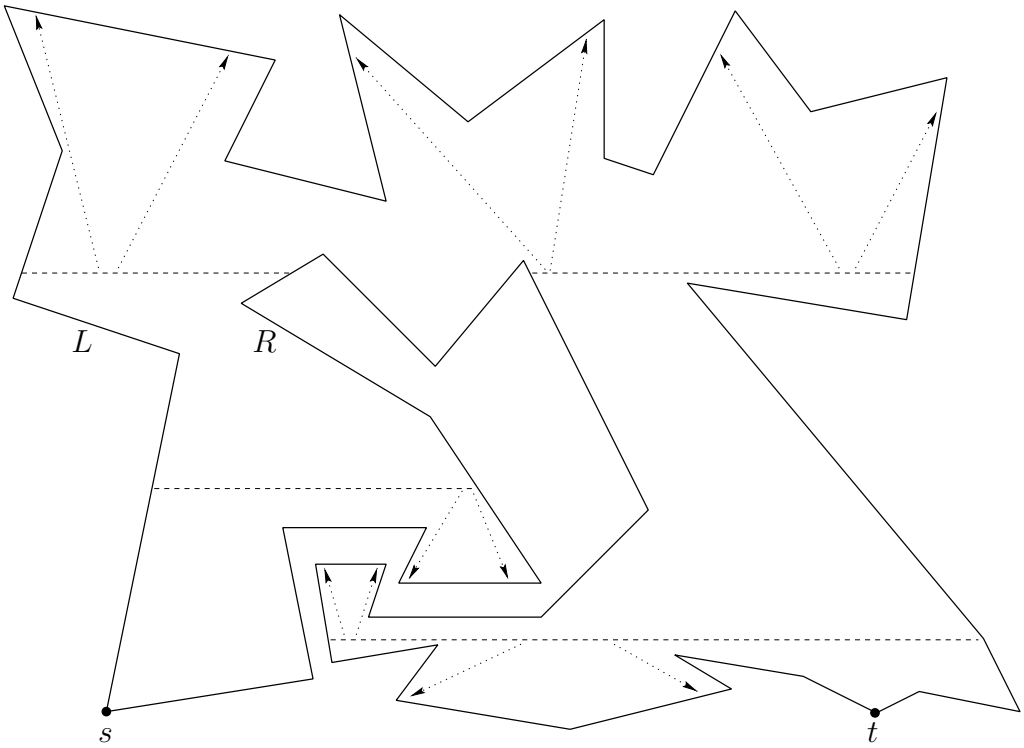

Fig. 1. A $\mathcal{G}$-street.

Lemma 4 *The class of $\mathcal{G}$-streets contains all streets and this containment is proper.*

Proof. Consider a street and a point p in its left chain L, see Figure 2. We want to show that p is visible from a point on an LR-chord. Within the street, p must be visible from some point $q \in R$. We shoot a horizontal ray from p into the interior of the street. If it hits R then p lies on an LR-chord and we are done.

Otherwise the line segment $\overline{pq}$ is not horizontal, and we move a point r from p to q along $\overline{pq}$ from which we continue to shoot horizontal rays in both directions, see Figure 2. We stop at the first position where a ray touches the right chain R, which must eventually happen since $q \in R$. This ray determines an LR-chord from which p is visible.

This proves that every street is a $\mathcal{G}$-street. In Figure 1, we have seen an example of a $\mathcal{G}$-street which is not a street. □

The following lemma states a simple property of LR-chords.

Lemma 5 *Every path in P from s to t intersects all LR-chords of the polygon.*

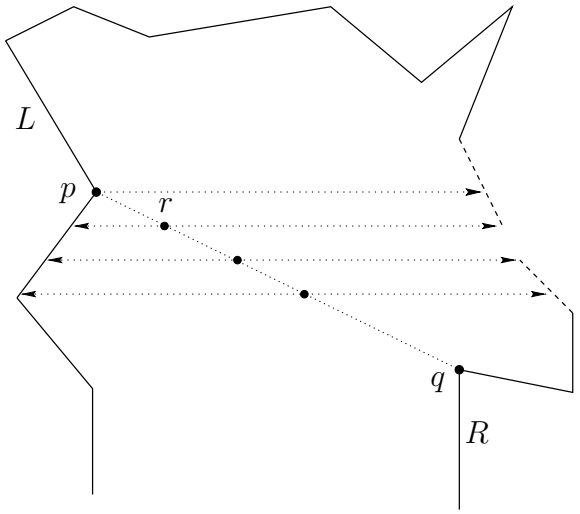

Fig. 2. A street is always a $\mathcal{G}$-street (Lemma 4).

Proof. Consider an LR-chord c and suppose c touches the chains L and R at the points p and q respectively. The line segment $\overline{pq}$ divides the polygon into two parts. The point t cannot be in the part which also contains s, otherwise p and q would be on the same chain L or R. This implies that s and t are in different parts and any path from s to t must intersect $\overline{pq}$. □

3 A strategy for rectilinear $\mathcal{G}$-streets

From now on, we concentrate on rectilinear $\mathcal{G}$-streets; all edges are either horizontal or vertical, see Figure 3 for an example. It is clear that the class of rectilinear $\mathcal{G}$-streets includes all rectilinear streets and is strictly larger.

Definition 6 • A horizontal edge of the polygon is called a *cut edge* if both of its vertices are reflex. The chord of a cut edge is called a *cut chord*.
- A horizontal edge is called a *step edge* if one of its vertices is convex and the other is reflex. The chord of a step edge is called a *step chord*.
- A cut chord divides the polygon into three parts, a step chord divides it into two parts. We call these parts the *regions* of the chord.

We refer to a cut chord or step chord as a *critical chord*, see Figure 4 for examples. It is easy to see that iff the two points s and t are in two different regions of a chord then it is an LR-chord.

Definition 7 A reflex vertex l is called a *landmark with respect to an LR-chord c* if

- l is visible from some point of c, and

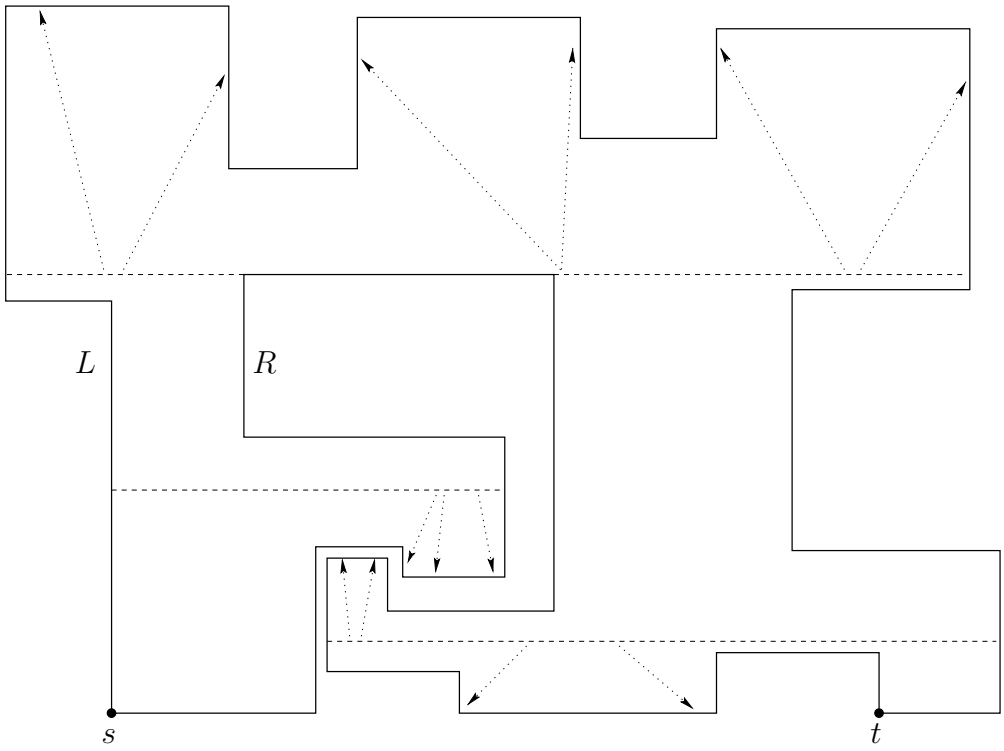

Fig. 3. A rectilinear $\mathcal{G}$-street.

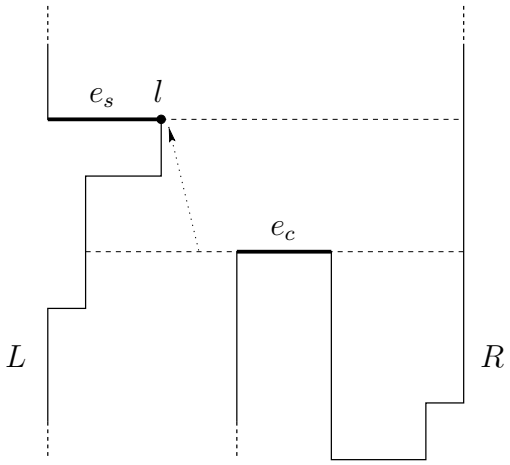

Fig. 4. A *step edge*, e_s, a *cut edge*, e_c, and the respective chords. Vertex l is a *landmark* with respect to the chord of e_c.

- both c and the vertical edge incident on l are part of the same region of the chord of l.

See Figure 4 for an example of a landmark. A landmark is always an endpoint of a cut chord or a step chord. Thus, for the robot to see a landmark will be an evidence that the target is hidden beyond the chord of that landmark.

Lemma 8 *Let l be a landmark with respect to an LR-chord and let c be the chord of the horizontal (cut or step) edge of l. Then the following is true.*

(1) The points s and t are in different regions of c, i. e. c is an LR-chord.
(2) All landmarks (if any) with respect to c are in the regions of c containing s and t.
(3) Either the point t or some landmark in the region of t is visible from some point on c.

Proof. We prove the lemma for the case of c being a cut chord. The proof for a step chord is similar.

Suppose, c is a cut chord which passes through the cut edge e. Suppose, the vertex v is adjacent to e and v is a landmark with respect to an LR-chord c_1. First, assume that s and t are in the same region of c. If s and t are in the region which does not contain c_1, then c_1 is not an LR-chord. If they are in the same region which contains c_1, then there is at least one point on the edge e which is not visible from any LR-chord and the polygon is not a $\mathcal{G}$-street, a contradiction. Hence, c is an LR-chord and the two points s and t are in two different regions of the chord c.

For proving the second statement, assume that there is a landmark l_2 on a horizontal edge e_2 in a region R_2 other than the ones which contain s and t. Note that no horizontal line segment (except the chord c) whose end points touch the polygon in the region R_2 can be an LR-chord. This is because the complete polygonal chain which constitutes this region belongs either to the L or to the R chain. So, there is at least one point on e_2 which is not visible from any LR-chord. This means the polygon is not a $\mathcal{G}$-street, a contradiction.

For the third statement, consider the two regions which do not contain the point s. We have already proved that t must be in one of these two regions. So, t can be hidden from all the points of the chord c only through the presence of a landmark. □

Corollary 9 *If c is the chord of the starting point s then the following holds.*

- *The point t or some landmark in the region of c containing t is visible from some point on c.*
- *The only visible landmarks are in the region of c containing t.*

Now, we turn to the description of our algorithm. We assume that at every position in its path, the robot can get the visibility map of the polygon through its sensors. The idea for our strategy is inspired by the strategy, called *doubling*, for searching a point on a line by Baeza-Yates et al. [2]. There, the robot goes

back and forth on the line, at each step doubling the distance to the start point, until the target is reached.

Theorem 10 [2] *The* doubling *strategy for searching a point on a line has a competitive factor of* 9*, and this is optimal.*

For the problem of finding a path in a $\mathcal{G}$-street, we apply a slight variant of the doubling strategy. Standing on a critical LR-chord, our robot performs doubling on this chord until the target becomes visible or a landmark is in sight. If at any time during this doubling a vertical wall is hit, the robot walks straight in the other direction. Also, it may happen that after reaching a cut or step chord, the robot immediately sees the next landmark and it does not execute doubling on this chord. We do not differentiate between all these variants of the doubling algorithm in our strategy and call all of them *doubling*.

The doubling strategy in [2] is formulated only for integer distances, such that the minimum distance to the target is 1. For distances arbitrarily close to 0 it is not competitive, strictly speaking, because the first step may be to the wrong direction. One can remedy this by introducing an additive constant in the definition of competitiveness.

Although we use real coordinates in our environment, this problem does not occur since we always know a lower bound for the distance to the target. At the start of each doubling, we chose the distance of the actual position of the robot to the nearest projection of all visible vertices on the actual chord as length of the first step.

Our algorithm, called *doubleX*, is presented (in pseudo-code) in Figure 5. The robot walks on a rectilinear path and tries to keep close to an L_1-shortest path. Initially, when the robot is at the point s, it either sees a landmark or the goal by executing the doubling algorithm on the chord of s. This is possible due to Corollary 9. So, initially the robot can choose a region depending on this landmark. From one sequence of doubling steps to the next, the robot moves from one LR-chord defined by a landmark to another LR-chord defined by the next landmark. These landmarks can be correctly chosen due to Lemma 8. If there is more than one visible landmark then the robot is free to choose one of them. In this process, ultimately the goal is reached.

We need the following lemma before proving the feasibility of step 10 of the algorithm.

Lemma 11 *If a landmark l is visible from the present position, the robot can reach the L_1-nearest point on the chord of l by an xy-monotone, i. e. L_1-shortest, path.*

```
1  procedure doubleX
2    s : the starting point.
3    t : the target point.
4    m: a horizontal chord.
5  begin
6    Let m be the chord of s.
7    Execute doubling on m until a landmark l or the target t is visible.
8    while (target point t is not visible) do
9      Let m be the chord of l.
10     Go to the L1-nearest point of m on an xy-monotone path.
11     Determine the region R of m where we come from.
12     Execute doubling on m
13       until a landmark l or the target t is visible in a region other than R.
14   end while
15   Go to t on an xy-monotone path.
16 end
```

Fig. 5. Strategy *doubleX* for rectilinear $\mathcal{G}$-streets.

Proof. We assume w.l.o.g. that the landmark l is on the right chain R and to the right and above the present position p of the robot, see Figure 6. The other cases are similar.

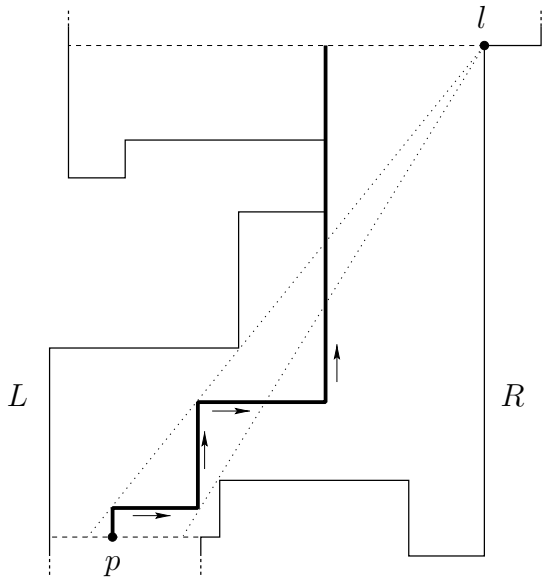

Fig. 6. An xy-monotone step from one LR-chord to another.

The robot goes vertically upwards from its present position until it reaches the chord of the landmark l (then we are done), or l becomes invisible. The visibility ray to l can only be obstructed by a reflex vertex of L. At this point, the robot starts going to the right until it is vertically below a point of the chord of l, or l becomes invisible again (whatever comes first).

This process is repeated until the chord of l is reached, see the thick line in Figure 6. Clearly, the resulting path is xy-monotone and its expansion in y-direction equals the vertical distance between the chords of p and of l.

The reached point on the chord of l either is vertically above p, or has an x-coordinate equal to the maximum x-coordinate of the subchain of L between the chord of p and the chord of l. Thus every path from p to a point of the chord of l expands in x-direction at least as much as our path does. This shows that we reach the chord of l at the L_1-nearest point from p. □

It is easy to see that if t is visible from the present position of the robot, it can reach t in a similar way, so that step 15 of the algorithm is feasible.

Lemma 12 *Starting from the point s, the robot correctly reaches the point t by executing procedure doubleX.*

Proof. Initially, there are two possibilities at the point s. If the robot sees the point t, it can go to t by a method similar to that in Lemma 11. Otherwise, it finds a landmark according to Corollary 9. From Lemma 5, the path should cross the chord which passes through this landmark. So, initially, the robot chooses a correct vertical direction. In every subsequent step, the robot reaches a critical chord. While executing the doubling algorithm at each such chord, it is always ensured from Lemma 8 that either the next landmark or the point t is found. As soon as the point t is visible, the robot can go to t by an xy-monotone path as mentioned before. □

We now estimate the length of the path generated by the robot.

Definition 13 The *reduced path* is a rectilinear path from s to t which only includes from the robot's path

- the xy-monotone motion from one critical chord to the next, and
- the horizontal step from the point where a critical chord is first reached to the point where it is left.

In other words, for the reduced path, we do not consider the extra movements generated due to the execution of the doubling algorithm.

Lemma 14 *The reduced path is an L_1-shortest path from s to t.*

Proof. Along its way from s to t, the robot stops at a sequence of critical chords $c_0, \ldots, c_i, \ldots, c_k$, where c_0 is the chord of s, and c_k is the last chord the

robot crosses before reaching t.

By induction, assume that the path is a shortest path up to the chord c_i. Clearly, this is true for $i = 0$. Now, we have to prove that the path up to c_{i+1} is shortest. If $i = 0$ or if the reduced path from c_{i-1} to c_{i+1} is xy-monotone, this follows from the induction hypothesis.

Otherwise, we have two possible cases. If the reduced path changes its vertical direction at c_i ("up" from c_{i-1} to c_i and "down" from c_i to c_{i+1} or vice versa) then c_i is a cut chord, and every shortest path from s to t goes along the edge of c_i.

If, on the other hand, the horizontal direction changes, then the path from c_{i-1} to c_i must touch a vertical edge at an extreme position such that every shortest path also touches that edge, since otherwise the robot's path, from c_{i-1}, would not reach the L_1-nearest point on c_i.

From this we conclude that the reduced path is shortest up to t. □

Theorem 15 *Procedure* doubleX *achieves a competitive factor of* 9 *in the* L_1 *metric, and this is optimal.*

Proof. From Lemma 14, the reduced path of the robot is a shortest path. Compared to a reduced path, the only extra path segments the robot traverses are the segments generated due to doubling on a step or cut chord. Suppose, for such a chord c_i, the robot reaches and leaves c_i at the points p_1 and p_2 respectively. From Theorem 10, the robot traverses at most 9 times the length of the segment $\overline{p_1p_2}$ due to doubling. Hence, the claim follows.

On the other hand, no better competitive factor can be achieved. Figure 7 shows how we can produce a $\mathcal{G}$-street with many caves at integer positions where t can be in any of the caves, such that searching for t in such a $\mathcal{G}$-street is essentially equivalent to searching a point on a line, for which [2] proves a lower bound of 9. □

The lower bound of 9 also applies if we compare the robot's path to the L_2-shortest path. But what is the competitive factor of our strategy in this case? Any L_1-shortest path is at most $\sqrt{2}$ times longer than the corresponding L_2-shortest path. From this one can conclude that our strategy *doubleX* achieves a competitive factor of $9\sqrt{2}$ ($\approx$12.73) in the L_2 metric. But with a closer look to the details of our strategy, one can prove a much better bound.

Theorem 16 *Procedure* doubleX *achieves a competitive ratio of* 9.06 *in the* L_2 *metric, and this is also optimal.*

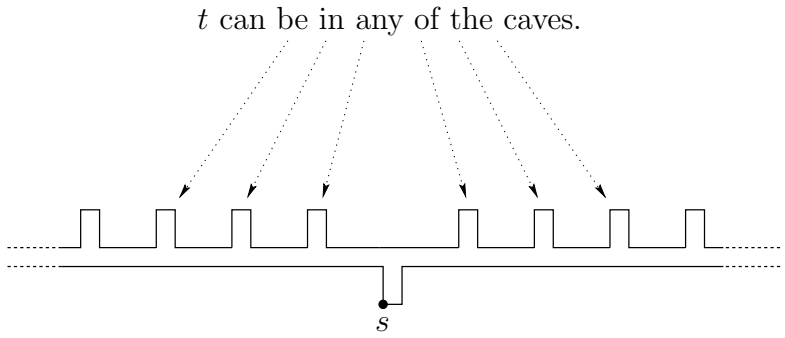

Fig. 7. Searching in a $\mathcal{G}$-street is at least as difficult as searching a point on a line.

Proof. We consider two reflex vertices p_1 and p_2 of the robots path such that the reduced path from p_1 to p_2 is xy-monotone but the reduced path from p_1 to the critical chord after p_2 (if it exists) is not xy-monotone, as well as the reduced path from the preceding chord of p_1 to p_2.

Let x_1 be the x-distance between p_1 and p_2 and y_1 the y-distance (such that the reduced path from p_1 to p_2 has length $x + y$).

The length of the L_2-shortest path from p_1 to p_2 is at least $\sqrt{x_1^2 + y_1^2}$. From the proof of Theorem 15 we know that the length of the robot's path is at most $9x_1 + y_1$. We may assume that $y_1 \neq 0$ and substitute $w = x_1/y_1$. The competitive factor must be less than the maximum value of

$$\frac{9x_1 + y_1}{\sqrt{x_1^2 + y_1^2}} = \frac{9w + 1}{\sqrt{w^2 + 1}}$$

for all possible values of w. Differentiation on the right hand side with respect to w shows that the maximum is reached at $w = 9$ at which

$$\sqrt{82} < 9.06$$

is the maximum value. The claim of the theorem follows by extending the analysis to every pair of such chords and from the fact that the L_2-shortest path must also visit the points p_1 and p_2.

López-Ortiz and Schuierer [24,26,29] remark that $\sqrt{82}$ is also a lower bound, so it is in fact the optimal competitive factor. To see this, one can adapt the proof of the lower bound in Theorem 15 to the L_2 case by introducing caves also to the lower part of the polygon in Figure 7 and by arranging the caves in bow-tie form with slope $\frac{1}{9}$. □

4 Conclusions

For the rectilinear case, we have extended the class of polygons for which competitive strategies are known for a robot, starting at a vertex s, to find a path to a vertex t with the help of the visibility map. Our strategy *doubleX* achieves optimal competitive factors of 9 in the L_1 metric, and of 9.06 in the L_2 metric.

So far, we have seen worst-case results for deterministic algorithms. However, it may be desirable to have randomized algorithms for our problem with a proven expected performance (competitive factor). Kao, Reif, and Tate [18] present an optimal randomized algorithm for searching on two paths with an optimal competitive factor of ≈ 4.59. Their method is very similar to *doubling*, they chose to multiply the lengths of subsequent steps by ≈ 3.59 instead of 2, the actual randomization happens only when the first step is determined as a random number.

Therefore, it is possible to make randomized variants of our algorithms just by replacing doubling by the method of [18]. We then get optimal expected performances of ≈ 4.59 in the L_1 metric, and ≈ 4.70 for the L_2 case.

But it should perhaps be noted that comparing expected and worst-case performances is somehow unfair ("Our randomized algorithm gives expected performance that is almost twice as good as is possible with a deterministic algorithm", from [18]). To be more precise, one should say that the *expected* performance of doubling (using multiplication factor 2) is about ≈ 5.33, which is not so far from the optimal ≈ 4.59, whereas the worst-case performance using multiplication factor ≈ 3.59 (from the optimal randomized algorithm) is ≈ 10.95, compared to the optimal 9.

Recent results [5,23,24,28] show that there are different, and larger classes of polygons that permit competitive searching, albeit with a bigger competitive factor. It is an interesting open question whether there is, in some sense, a largest class of polygons which can be searched competitively.

Acknowledgement. We would like to thank the anonynous referees for their valuable comments.

References

[1] S. Albers, K. Kursawe, and S. Schuierer. Exploring unknown environments with obstacles. In *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pages 842–843, 1999.

[2] R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Inform. Comput.*, 106:234–252, 1993.

[3] A. Blum, P. Raghavan, and B. Schieber. Navigating in unfamiliar geometric terrain. *SIAM J. Comput.*, 26(1):110–137, 1997.

[4] G. Das, P. Heffernan, and G. Narasimhan. LR-visibility in polygons. *Comput. Geom. Theory Appl.*, 7:37–57, 1997.

[5] A. Datta, C. A. Hipke, and S. Schuierer. Competitive searching in polygons: Beyond generalised streets. In *Proc. 6th Annu. Internat. Sympos. Algorithms Comput.*, volume 1004 of *Lecture Notes Comput. Sci.*, pages 32–41. Springer-Verlag, 1995.

[6] A. Datta and S. Soundaralakshmi. Motion planning in an unknown polygonal environment with bounded performance guarantee. In *Proc. 1999 IEEE Internat. Conf. Robot. Autom.*, 1999.

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

[8] G. Dudek, K. Romanik, and S. Whitesides. Localizing a robot with minimum travel. *SIAM J. Comput.*, 27(2):583–604, 1998.

[9] S. K. Ghosh and S. Saluja. Optimal on-line algorithms for walking with minimum number of turns in unknown streets. *Comput. Geom. Theory Appl.*, 8(5):241–266, 1997.

[10] L. J. Guibas, R. Motwani, and P. Raghavan. The robot localization problem. *SIAM J. Comput.*, 26(4):1120–1138, 1997.

[11] F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. A competitive strategy for learning a polygon. In *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, pages 166–174, 1997.

[12] F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. The polygon exploration problem: A new strategy and a new analysis technique. In *Robotics: The Algorithmic Perspective, Proc. 3rd Workshop Algorithmic Found. Robot.*, pages 211–222. A. K. Peters, 1998.

[13] C. Icking and R. Klein. The two guards problem. *Internat. J. Comput. Geom. Appl.*, 2(3):257–285, 1992.

[14] C. Icking and R. Klein. Competitive strategies for autonomous systems. In H. Bunke, T. Kanade, and H. Noltemeier, editors, *Modelling and Planning for Sensor Based Intelligent Robot Systems*, pages 23–40. World Scientific, Singapore, 1995.

[15] C. Icking and R. Klein. Searching for the kernel of a polygon: A competitive strategy. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 258–266, 1995.

[16] C. Icking, R. Klein, and E. Langetepe. An optimal competitive strategy for walking in streets. In *Proc. 16th Sympos. Theoret. Aspects Comput. Sci.*, volume 1563 of *Lecture Notes Comput. Sci.*, pages 110–120. Springer-Verlag, 1999.

[17] C. Icking, A. López-Ortiz, S. Schuierer, and I. Semrau. Going home through an unknown street. Technical Report 228, Department of Computer Science, FernUniversität Hagen, Germany, 1998.

[18] M.-Y. Kao, J. H. Reif, and S. R. Tate. Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. *Inform. Comput.*, 133(1):63–79, 1996.

[19] O. Karch, H. Noltemeier, and T. Wahl. Robot localization: Theory and implementation. In *Abstracts 13th European Workshop Comput. Geom.*, pages 17–19. Universität Würzburg, 1997.

[20] R. Klein. Walking an unknown street with bounded detour. *Comput. Geom. Theory Appl.*, 1:325–351, 1992.

[21] J. M. Kleinberg. On-line search in a simple polygon. In *Proc. 5th ACM-SIAM Sympos. Discrete Algorithms*, pages 8–15, 1994.

[22] J.-H. Lee and K.-Y. Chwa. Tight analysis of a self-approaching strategy for the online kernel-search problem. Technical report, Department of Computer Science, KAIST, Taejon, Korea, 1998.

[23] J.-H. Lee, C.-S. Shin, J.-H. Kim, S. Y. Shin, and K.-Y. Chwa. New competitive strategies for searching in unknown star-shaped polygons. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pages 427–429, 1997.

[24] A. López-Ortiz. *On-line target searching in bounded and unbounded domains.* PhD thesis, Univ. Waterloo, Waterloo, Canada, 1996.

[25] A. López-Ortiz and S. Schuierer. Going home through an unknown street. In *Proc. 4th Workshop Algorithms Data Struct.*, volume 955 of *Lecture Notes Comput. Sci.*, pages 135–146. Springer-Verlag, 1995.

[26] A. López-Ortiz and S. Schuierer. Generalized streets revisited. In *Proc. 4th Annu. European Sympos. Algorithms*, volume 1136 of *Lecture Notes Comput. Sci.*, pages 546–558. Springer-Verlag, 1996.

[27] A. López-Ortiz and S. Schuierer. Walking streets faster. In *Proc. 5th Scand. Workshop Algorithm Theory*, volume 1097 of *Lecture Notes Comput. Sci.*, pages 345–356. Springer-Verlag, 1996.

[28] A. López-Ortiz and S. Schuierer. Position-independent near optimal searching and on-line recognition in star polygons. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pages 445–447, 1997.

[29] A. López-Ortiz and S. Schuierer. Lower bounds for searching on generalized streets. University of New Brunswick, Canada, 1998.

[30] V. J. Lumelsky and A. A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.

[31] C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoret. Comput. Sci.*, 84(1):127–150, 1991.

[32] S. Schuierer. Efficient robot self-localization in simple polygons. In *Abstracts 13th European Workshop Comput. Geom.*, pages 20–22. Universität Würzburg, 1997.

[33] S. Schuierer and I. Semrau. An optimal strategy for searching in unknown streets. In *Proc. 16th Sympos. Theoret. Aspects Comput. Sci.*, volume 1563 of *Lecture Notes Comput. Sci.*, pages 121–131. Springer-Verlag, 1999.

[34] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28:202–208, 1985.

[35] L. H. Tseng, P. Heffernan, and D. T. Lee. Two-guard walkability of simple polygons. *Internat. J. Comput. Geom. Appl.*, 8(1):85–116, 1998.