Optimal Strategy for Walking in Streets with Minimum Number of Turns for a Simple Robot

Azadeh Tabatabaei and Mohammad Ghodsi

Department of Computer Engineering, Sharif University of Technology
atabatabaei@ce.sharif.edu
Sharif University of Technology and School of Computer Science, Institute for Research in Fundamental Sciences (IPM)
ghodsi@sharif.edu

Abstract. We consider the problem of walking a simple robot in an unknown street such that number of the robot turns is as small as possible. The robot has a minimal sensing capability that can only report the discontinuities in the depth information (gaps), and location of the target point once it enters in its visibility region. We present an online strategy that generates a search path for the robot, based on the location of gaps, to reach the target t, starting from s. Although the path coincides with the previously known search path, the confined sensing model is interesting in its own. In the previous strategy a robot that has access to the map of its visibility region that has seen so far, explores the street. The robot, based on the location of gaps and changes in its visibility region boundary, searches the street while our simple robot cannot sense the visibility region. Furthermore, we have proposed a search strategy that generates a rectilinear path for the robot to reach the target in a rectilinear street with optimal number of turns.

Keywords: Computational Geometry . Minimum Link Path . Simple Robot . Street Polygon . Unknown Environment .Competitive Ratio

1 Introduction

Due to many real life applications, path planning in unknown environments is considered as a fundamental problem in robotics, computational geometry and online algorithms [2, 3, 19]. A robot based on the information gathered from its tactile sensors moves in the environment until it achieves its target. Neither the geometric map of the environment nor the location of the target point are known to the robot. The volume of information provided to the robot depends on the strength of its sensors. Employing a simple robot with a simple sensing model has many advantages such as: low cost of hardware, being applicable to many situations, and being robust against sensing uncertainty and noise [1, 5, 12, 14, 17].

Here a simple robot with an abstract sensor that can only detect the order of depth discontinuities (or gaps) of the boundary in its visibility region is considered. Each discontinuity corresponds to a portion of the environment that is not visible to the robot (Fig. 1). The gap sensor assigns a label of L or R to each gap g depending on which side of the gap the hidden region is. The robot tracks the directions of the gaps and can rotates as often a critical event that changes topologically location of the gaps occurs. These events are: appearance, disappearance, merging, and splitting of gaps. Moving along straight lines is cheap, but rotation is expensive for the robot [15]. Also, the robot recognizes a target point t as it enters in its omnidirectional and unbounded field of view.

The robot using the information gathered through the sensor starts navigating a street environment from a start point s to reach a target t. A street is a simple polygon P with two vertices s and t such that the counter-clockwise polygonal chain R_{chain} from s to t and the clockwise one L_{chain} from s to t are mutually weakly visible. In other words each point on the left chain is visible from at least one point on the right chain and vice versa [6], (Fig. 1.a). Note that minimizing the number of turns is an essential criterion in path planning for such robot. This problem is also known as the shortest path problem in the link metric, in some literatures [8, 10].

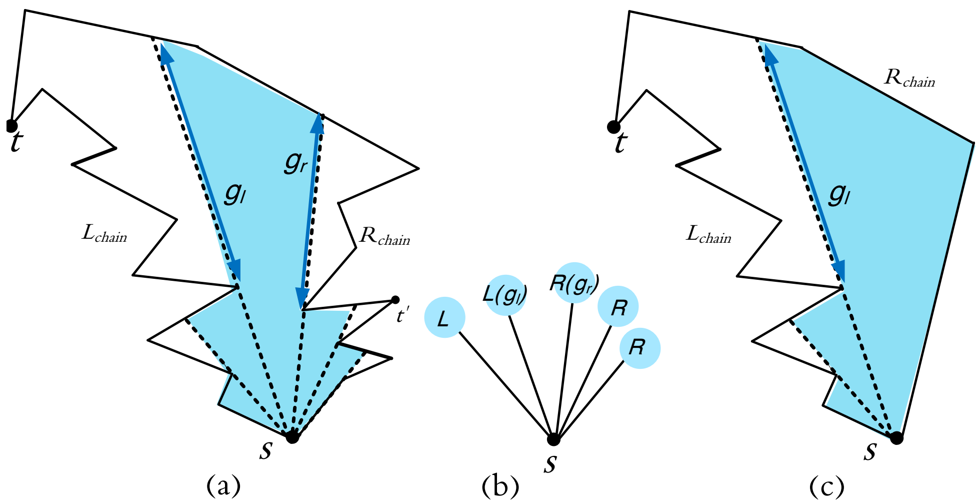

Fig. 1. (a) A street in which L_{chain} is the left chain and R_{chain} is the right chain. The colored region is the visibility polygon of the point robot at the start point s. (b) The position of discontinuities in the depth information detected by the sensor at the start point in (a). (c) A street that has only left gaps at the start point.

The problem of walking in unknown streets with minimum number of turns was first studied by Ghosh and Saluja [7]. They presented an optimal online algorithm for a robot with an on-board vision system. Throughout the search path, the robot maintains the map of the portion of the street that it has seen so far. The robot, based on changes of the gaps and essentially based on the changes of its visibility region boundary, decides to turn at each moment.

In contrast with the robot, our simple robot cannot distinguish the visibility boundary. The robot, only based on the critical events that changes the locations of the gaps, should track the direction of some gaps to reach the target. Without maintaining the map of the visibility region of the environment, only by memorizing the location of two essential gaps, in a constant space, we present an online search strategy for the simple robot. We show, even with such the confident sensing model, the generated path by the strategy coincides with the path that Ghosh and Saluja robot tracks. So, the competitive ratio (the worst-case ratio of number of turns in the robot search path to the number of turns in the minimum link path from s to t.) of our strategy is $2 - 1/m$, as well as the robot employed in [7]; m is the link distance between s and t. Also, we prove a competitive ratio of $1 + 1/m$ for searching a rectilinear street.

Related Works: The simple robot system applied in this paper was first presented by Lavalle *et al.* [18]. They proposed Gap Navigation Tree (GNT) as a mean to maintain location of the gaps sensed by the robot for navigating the unknown scene. GNT is used for solving many visibility problems in unknown environment [9, 11, 17]. A competitive strategy is presented for walking in streets for a point robot that is equipped with the gap sensor in [13, 14] such that the generated path is at most 11 times longer the shortest path.

Other minimal sensing models have been introduced by other researchers. Suri *et al.* [12] offered a robot system in which the robot can only sense the combinatorial (non-metric) properties of its surroundings. The sensor detects the vertices of the polygon in its visibility region, and can distinguish if there is an edge between consecutive vertices of the region. Then, the robot is empowered by Disser*et al.*; it is equipped with a compass [1]. Katsev *et al.* [5] introduced a simple robot that performs wall-following motions and can traverse the interior of the scene only by following a direction that is parallel to an edge of the environment. Despite of the minimal capabilities, all of them have shown that their robot can provide many geometric reasoning and executes many non-trivial tasks such as counting vertices, solving pursuit-evasion problems and mapping a polygon.

2 The Sensing Model and Motion Primitive

A point robot starts exploring an unknown street until the target t is achieved, starting from s. The robot is equipped with a sensor that detects each discontinuities in depth information that referred as gaps. The sensor reports a cyclically ordered location of the gaps in its visibility region, (Fig. 1). Also, the robot allocates a label of L or R (left or right) to each gaps. Each label shows direction of the hidden region behind a gap relative to the robot's heading [17]. The robot can only track the gaps and records their topological changes. These changes are: appearance, disappearance, merging, and splitting of gaps. The appearance and disappearance events happen when the robot crosses the inflection rays, (Fig.

2). Each appearance event generates a gap that corresponds to a portion of the environment that was so far visible, and now is invisible. A gap that is generated by an appearance event, during the movement, is called a primitive gap and the other gaps are non-primitive gaps. The merge and split events occur when the robot crosses the bitangent complements, (Fig. 2).

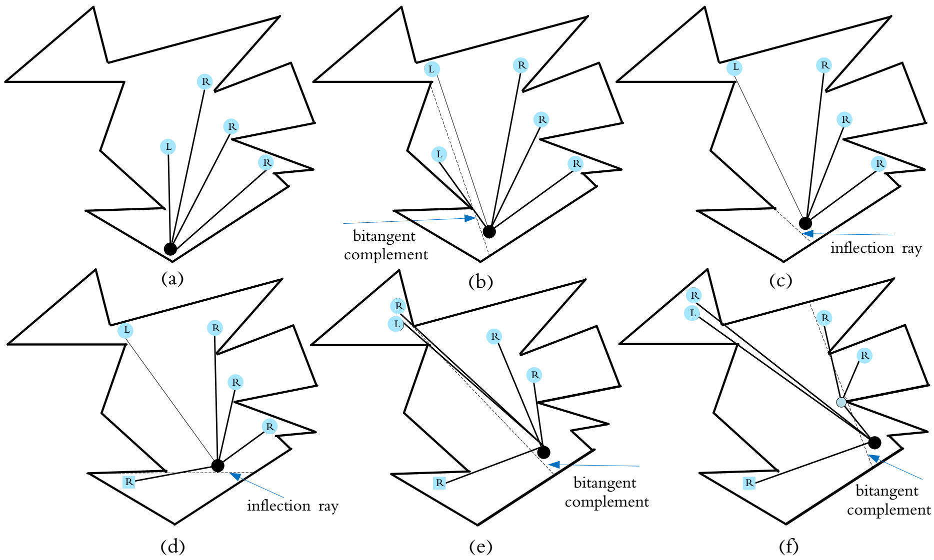

Fig. 2. Illustration of the dynamically changes of the gaps as the robot moves towards a gap. The dark circle denotes the location of the robot, and squares and other circles display primitive and non-primitive gaps respectively. (a) Existing gaps at the beginning. (b) A split event. (c) A disappearance event. (d) An appearance event. (e) Another split event. (f) A merge event.

In order to cover entire region, the robot follows the non-primitive gaps; the region hidden behind the primitive gaps has already been covered. The robot moves along a straight line towards non-primitive gaps, and may rotate as a critical event occurs. Also, the robot makes a turn when a wall of the environment is hit. As the target enters in the robot's visibility region, the robot orients its heading with the target, and walks towards it. Note that moving along straight lines is cheap, but rotation is expensive. At the point in which there is no non-primitive gap the entire environment has been observed by the robot.

3 Definitions and Preliminary Results

In this section, we briefly iterance some essential properties, and studied facts of gaps in streets, from [6, 14]. Also, we describe a necessary feature of gaps which aids the simple robot to perceive the changes of visibility boundary, in walking. Note that the robot cannot potentially recognize the visibility boundary and the visibility region. At each point of the search path, the sensor detects the target t or reports a set of gaps with the label of L or R (l-gap and r-gap for abbreviation).

Definition 1. *[14] In the set of non-primitive l-gaps, the gap which is in the right side of the others is called most advanced left gap and is denoted by g_l. Analogously, in the set of non-primitive r-gaps, the gap which is in left side of the others is called the most advanced right gap and is denoted by g_r, (Fig. 1.b).*

The two gaps have a fundamental role in path planning for the robot, such that:

Lemma 1. *[6, 14] On any point of the robot search path, the target t is hidden behind one of the most advanced gaps unless t is visible to robot.*

From Lemma 1, the target is constantly behind the most advanced gaps. So, at the start point s, there exists at least one of g_l and g_r unless t is visible from s. The case in which both of advanced gaps exist is called a funnel case [4, 13]. During the walking, whatever memorized is only location of g_l and g_r compared to Ghosh and Saluja [7] robot that memorizes everything which has been visible so far. The robot based on the changes of visibility region boundary (functional ray) and gaps explores the environment while our simple robot has no access to the visibility boundary.

As the robot moves, each of the critical events (appearance, disappearance, merge, and split) that change the locations of reported gaps by robot's sensor may dynamically change g_l and g_r, as follows:

1. **Split Event**
   When g_r/g_l splits into g_r/g_l and another r-gap/l-gap, then g_r/g_l will be replaced by the r-gap/l-gap, (point 1 in Fig. 3). We refer to this event as split 1.
   When g_r/g_l splits into g_r/g_l and another l-gap/r-gap, then g_l/g_r will be replaced by the l-gap/r-gap, (point 2 in Fig. 3 and point 2 in Fig. 4.a). We refer to this event as split 2.
2. **Appearance Event**
   Each appearance event generates a gap that hides a portion of the street that already was visible. Such gap is a primitive gap. So, this event does not update g_l and g_r, (point 3 in Fig. 3).
3. **Merge Event**
   When g_l or g_r merges with another gap, location of this gap will be memorized as an address to g_l or g_r, (point 4 in Fig. 3).
4. **Disappearance Event**
   As soon as each of g_l or g_r disappears, it will be eliminated from the memorized data for robot path planning, (point 5 in Fig. 3 and point 1 in Fig. 4.b).

By the above discussion, the events which change the location of g_l or g_r are: split 1, split 2, merge, and disappearance.

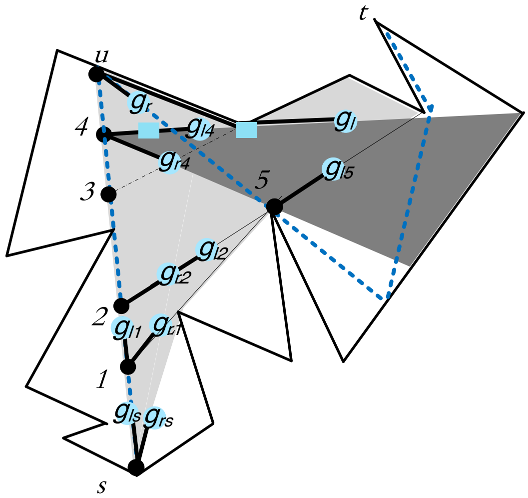

Fig. 3. g_{rs} and g_{ls} are the most advanced gaps at the start point s. g_{ri} and g_{li} are the most advanced gaps at point i. The dotted path that connects s to t is the robot search path. At each point i, an event arises. Light gray region shows the funnel from s to point 2 that changes at point 2. Dark gray region is the funnel at point 4.

Now we present an important attribute of g_l and g_r from which the simple robot can find out the changes of visibility boundary, as often as each of the critical event arises.

Definition 2. *We refer to g_l or g_r as an active gap if further movement along the selected direction allows the robot to see more of the hidden region behind the gap.*

The important key is that the simple robot how can distinguish if a gap is active or not from g_l or g_r position.

Lemma 2. *At the start point the simple robot can select a direction such that g_r and g_l are active.*

Proof. In the first case, one of the two gaps exists; for example g_r. Moving along each direction which is on or left side of g_r increases the robot visibility region. Since the robot can only move along the direction of gaps, g_r is active if and only the robot moves toward it. The situation in which only g_l exists is symmetric, and it will be an active gap when the robot moves toward it, analogously.
In the funnel case, moving along each direction which is on or between g_r and g_l increases the robot visibility region. So, by moving toward each of g_r and g_l, both of them will be set as an active gap.

As the events (split 1, split 2, merge, disappearance) arise, g_r and g_l may switch from being active to being inactive.

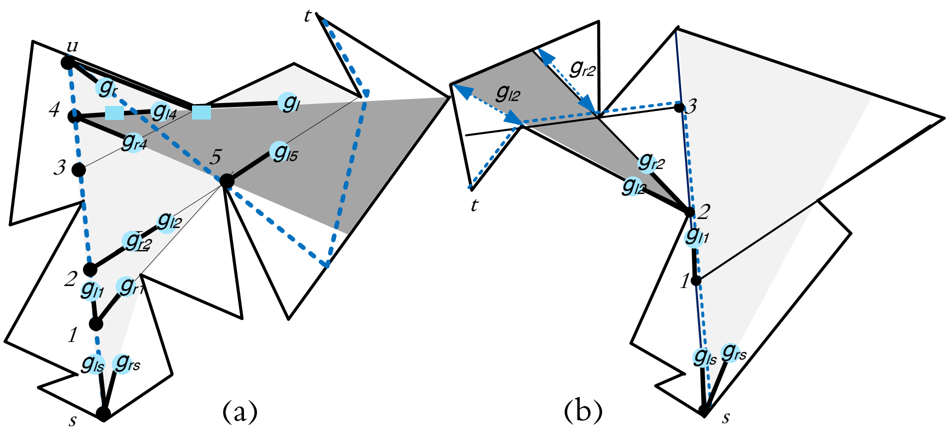

Fig. 4. The dotted path is the robot search path from s to t. Gray regions show the funnels. (a) Dark gray region is the funnel at point 4 (b) Dark gray region is the funnel at point 2.

Lemma 3. *Within walking along a selected direction our simple robot can detect switch of a gap from being active to inactive, without sensing the visibility region.*

Proof. First consider the case in which only one of g_r and g_l exists, for example only g_l. A disappearance event clears the hidden region behind the gap. A split 1 event updates g_l while it will remain active. By a split 2 event a funnel situation starts via splitting g_l into g_l and an r-gap. The most advanced gaps of this funnel are g_l and the r-gap. So the r-gap is current g_r. Current direction is not between the two gaps, but it is right side of g_l. So, g_r is set as inactive gap, and g_l remains active, (point 2 in Fig. 4.a). A merge event cannot arise by moving towards g_l. The situation in which only g_r exists is symmetric, and an inactive g_l may be generated analogously.
In a funnel condition, a disappearance event clears the hidden region behind one of g_r and g_l, but the other remains as an active gap. A split 1 event updates g_l or g_r while the gaps remain active. By a split 2 event, a new funnel case is constructed (point 2 in Fig. 3). One of the most advanced gaps of this funnel belongs to the previous funnel, so this gap remains active. The other is revealed via splitting the gap of the previous funnel. Further movement along the current direction does not allow the robot to see more of the hidden region behind the gap. So, the gap is set as inactive gap, (gap g_l at point 4 in Fig. 3).
Note that by each merge event that merges an active gap with another gap, the active gap will switch to inactive gap.

4 Algorithm

Here we explain our strategy for leading the robot in a street to reach the target t, starting from s, such that the number of turns in the robot's search path is as small as possible. In contrast with the previous research, the visibility region is not available to the robot. Our strategy only based on the locations of gaps generates a search path for the simple robot.

From Lemma 1, the target is constantly behind the most advanced gaps. So, at the start point s, there is at least one of g_l and g_r unless t is visible from s. If only one of them exists, the robot moves towards the gap in order to cover the region that is behind it, (Fig. 1.c). When a funnel situation arises at the start point, the robot moves towards one of the most advanced gaps, for example g_l, to cover the region hidden behind it, (Fig. 1.a). Our main idea for reducing the number of turns (links) of the search path is maximum use of a selected direction.

The robot continues to walk along the selected direction and makes a turn as often as each of the following conditions occurs.

1. The robot hits a point u on a wall such that it cannot proceed further. Recall that the counter-clockwise polygonal chain from s to u or the clockwise one from s to u or both are weakly visible from each simple path that connects s to u [7]. So, by arising a disappearance event or a split 2 event, the most advanced gap that lies on the chain has become inactive before reaching the hit point u. At the hit point, the robot moves towards the gap that has not become inactive. Also at the turn point, the existing most advanced gaps will be set as active gaps again, (point u in Fig. 3).
2. The robot achieves a point in which non of g_r and g_l are active. Further movement along the current direction does not allow the robot to see more hidden region behind the gaps. One of the situations below has arisen.
   - The existing active gap, g_r or g_l, merges with an inactive gap (point 3 in Fig. 4.a). At this point, the robot turns towards the merged gap and the gap will be set as an active gap.
   - The existing active gap, g_r or g_l, disappears. At this point, the robot turns towards the existing gap. So, the gap will be set as an active gap.

5 Analysis of the Algorithm

Here we show that the simple robot, only using location of g_l and g_r, achieves the target t starting from s. Also we demonstrate that our robot can follow the same optimal path traversed by Ghosh and Saluja [7] robot while our robot is strongly weaker than the robot. Their robot maintains the map of the street that has seen so far, and explores the environment using of gaps location and especially by using of changes in its visibility region boundary.

Theorem 1. *Our search strategy using a constant memory space terminates while the target is attained.*

Proof. During the walking, the robot maintains the addresses of g_l and g_r. The robot always selects a direction to moves such that the hidden region behind at least one of the two gaps decreases. Even the address of g_l and g_r may update via split1, split2 and merge events, but the number of the events are finite; each corresponds to a crossing over a bitangent. Since the target is hidden behind one of the two gaps, it will be achieved in a finite time using a constant memory space.

In order to enumerate the number of the links in the search path, and prove a competitive ratio for our strategy, we remark the concept of eave, from [8]. An edge $u_i u_j$ of $SP(s,t)$ (Euclidean shortest path from s to t) is called an eave if u_{i-1} and u_{j+1} lie on the different sides of the line that connects u_i to u_j (Fig. 6).

Lemma 4. *At each turn points of the generated path by our strategy, both of the left tangent and the right tangent to $SP(s,t)$ lie inside the street when $SP(s,t)$ has no eave.*

Proof. Consider $SP(s,t)$ has only left turn, by our strategy, the robot select a direction that is right side of g_l. The robot turns as soon as g_l becomes inactive by a merge event, or by hitting a wall. Then the two tangents lie inside the street, point z_1 and z_2 in Fig. 5.a. When $SP(s,t)$ has only right turn is symmetric.

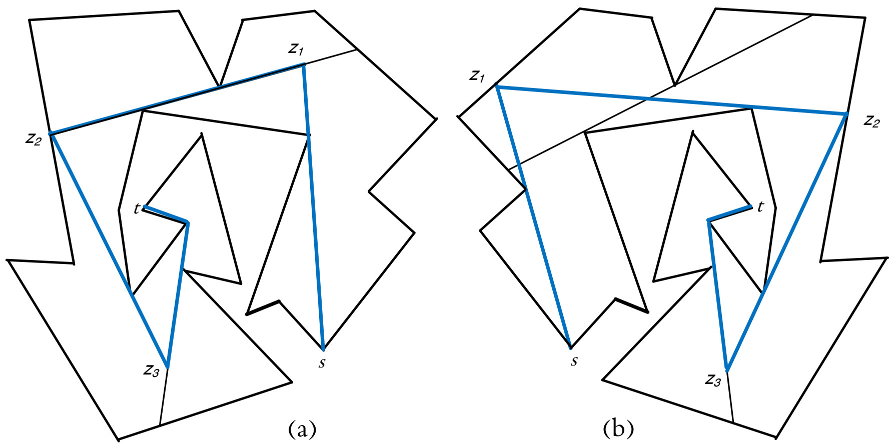

Fig. 5. The bold path is the robot search path. The left and right tangent from each turn point z_i lies inside the street. (a) Shortest path from s to t has only left turns. (b) Shortest path from s to t has only right turns.

Theorem 2. *The robot achieves the target t, starting from s, with at most $m + 1 + e$ links where m is the link distance between s and t and e is the number of eaves in $SP(s,t)$. Also, the competitive ratio is $2 - 1/m$.*

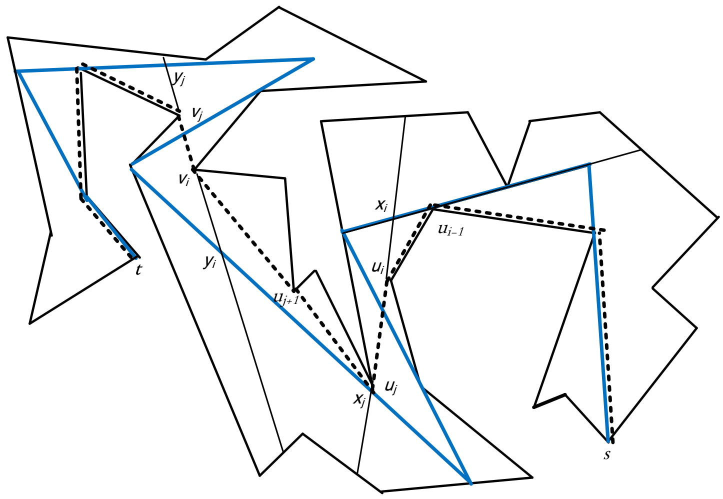

Fig. 6. The general case in which $SP(s,t)$, dotted path, has both left and right turns. Bold path is the robot search path.

Proof. In the special case when there is no eave in $SP(s,t)$, Lemma **??** shows that the our strategy generates a search path that coincides with the path that the robot in [7] traversed. So, the number of links in the path is at most $m+1$. In the general case, there are some eaves in the path. Let u_iu_j be the first eave of the path. The shortest path from s to u_i has only right turns or only left turns. If we extend the eave to the street boundary of both side, by our strategy, the robot achieves a point x_i on the first extension with at most one link more than the optimal path, see Fig. 6. Assume that the $SP(s,t)$ makes left turns from s to u_i and makes a right turn at u_j. The robot, after passing through the point x_i, turns left as soon as each of the conditions (1) or (2) for turning arises and crosses the eave. by our strategy, g_l becomes inactive before the robot achieves the next turn point. So, the robot turns towards the current g_r and crosses the other extension of the eave, x_j. Thus the robot traverses from first extension of an eave to the other extension of the eave with at most two links. Since there is a minimum link path that contains all eaves of $SP(s,t)$ [8] ($e \leq m-2$), the number of links in the path is at most $e+m+1$ and the competitive ratio is $2-1/m$.

lemma 4 and Theorem 2 show the trajectory generated by our strategy is same as the route for the robot in the previous research despite the weakness of our robot.

6 Walking Along a Rectilinear Path in Rectilinear Streets

Now, we consider a special version of the problem in which the simple robot has to follow a rectilinear path in a rectilinear street. The goal is to decrease the number of links in the path as much as possible. We show that our simple robot can achieve the target t, using at most $m + 1$ links, starting from s where m is the link distance between s and t. The number of links in the path is as small as the number of links that a robot with a complete vision of the environment requires to move from s to t.

We assume that the edges of the street are vertical or horizontal. Also, the robot carries a compass that denotes to it the north, west, south and east directions. The robot follows a vertical or horizontal direction while detecting the location of gaps, see Fig. 7.

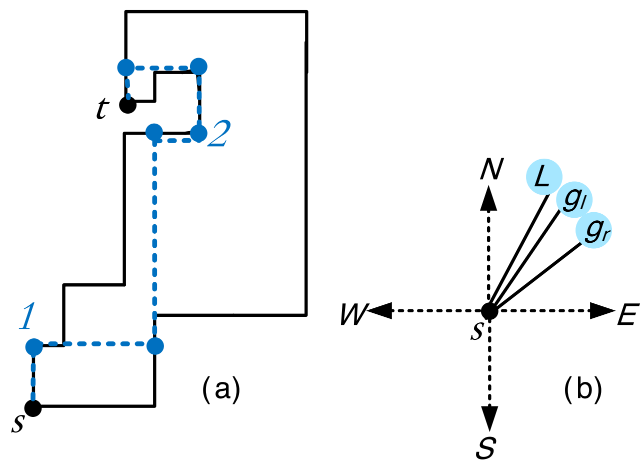

Fig. 7. (a) The dotted path is the robot search path in the rectilinear street. (b) The detected gaps and the compass direction at the start point s.

Whenever the target is not visible to robot, it is hidden behind one of g_l or g_r. So, the location of g_l and g_r that memorized during the navigation changes as explained in section 3. When the target is not visible to the robot, the possible locations of the two gaps are shown in Fig. 8, Fig. 9 and Fig. 10 [7, 13].

At the start point, the robot selects a direction to follow based on the locations of g_l and g_r:

1. When each of g_l or g_r is rectilinear (it is collinear with one of the compass directions), the robot moves towards the rectilinear gap (Fig. 8).
2. When one of the compass direction lies between g_l and g_r, the robot moves along the direction (Fig. 9.a).

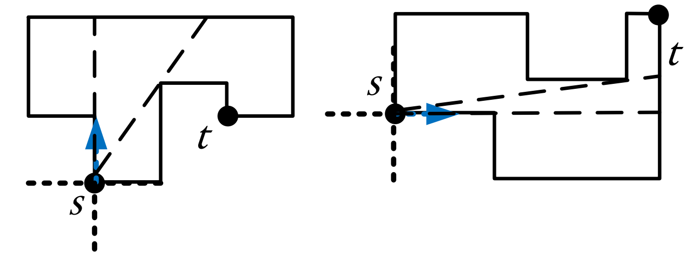

Fig. 8. There is a rectilinear gap, collinear with one of the compass direction, at the start point.

3. When only one of the two gaps exists. If the gap is g_l, the robot moves along the direction of the compass which lies clockwise after the gap. Analogously, if the gap is g_r, the robot moves along the direction of compass which lies counterclockwise after the gap (Fig. 10).
4. When the two gaps lie between consecutive directions of the compass, the robot moves towards one of the two directions (Fig. 9.b).

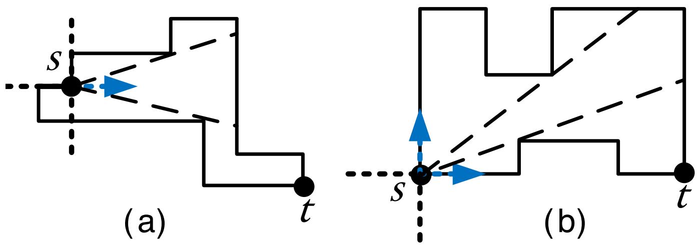

Fig. 9. The arrows show the directions that the robot may move along, by our strategy.

The robot continues to move along the selected direction as often as one of the following situations arises.

- The robot hits a boundary point. The robot, based on the current locations of the most advanced gaps, similar to the start point, selects a direction to move along. Note that this direction is orthogonal to the previous direction (point 1 in Fig. 7).
- The robot achieves a point in which one of the two gaps is rectilinear. In other words, its current direction is perpendicular to the direction of the gap. The robot turns towards the gap which is collinear with one of the compass directions (point 2 in Fig. 7).

The robot repeats the process until it reaches a point from which target t is visible. At the point, the robot moves along the current direction until the

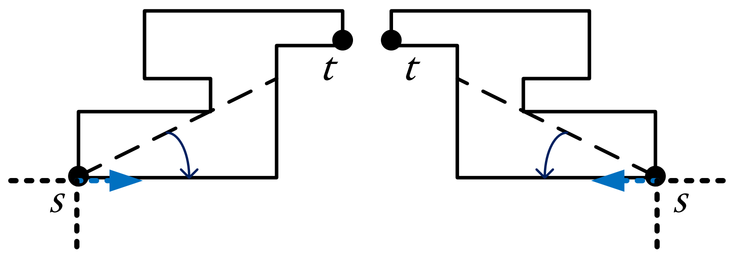

Fig. 10. The bold arrows show the directions that the robot moves along, at the start point.

target t becomes visible along one of the compass directions. Then, the robot turns towards the target. Now we show that our strategy generates a search path for the simple robot with at most $m + 1$ link where m is the orthogonal link distance between s and t.

Lemma 5. *There is a minimum link path that all of its links except its first link coincide with the robot search path.*

Proof. At the start point s, the robot based on the position of the most advanced gaps, selects a direction to move as explained in the strategy. Assume that the robot selects its first link, at the start point, under each of the conditions 1,2, and 3 in the strategy. Moving along the direction causes to decreases the hidden region behind the most advanced gaps (if the gaps exists). So, they are active gaps. Either the target t be hidden behind g_l or be hidden behind g_r, by moving along the selected direction, will be covered. Then, there is a minimum link path that its first link coincides with the robot path.

If condition 4 arises at the start point, two gaps lies in the same quadrant, the robot follows one of the rectilinear directions of the quadrant containing the gaps. During the movement one of the most advanced gap is active and the other is inactive. If the target is hidden behind the active gap, there is a minimum link path that its first link coincides with the robot path. Otherwise the robot takes an additional link. Throughout the searching process, the robot makes a turn as often as one of the most advanced gaps becomes rectilinear (same as condition 1 at the start point). Then, the robot will take at most one additional link over the minimum link path to reach the target t.

The Theorem below is the main result of this section.

Theorem 3. *Our strategy generates an orthogonal search path, for the robot, with a competitive ratio of* $1 + 1/m$*. Also, the Result is optimal.*

Proof. A similar argument to the proof of Theorem 1 shows the robot achieves the target. Also, Lemma 5 proves that competitive ratio of the strategy is $1+1/m$. Generating a path with $m + 1$ links for a robot with complete vision of the environment is optimal path [7]. So, this result is optimal for our simple robot.

7 Conclusions

In this paper,we studied the problem of walking in streets with minimum number of turns for a simple point robot. The robot has a minimal sensing model that can only detect the gaps and the target in the street. We proposed an online search strategy that generates a search path for such robot with optimal number of turns. The robot only by maintaining the locations of some gaps in a constant memory space traverses the street. The information that our robot uses to traverse is less than the information that the robot in previous research uses. In the former research, a robot that memorizes the region has seen so far, using the locations of gap and changes in its visibility boundary, explores the street. We demonstrated that, despite the weakness in our robot system model, the generated search path by our strategy coincides with the search path of the stronger robot. Also, for a special case in which the street is rectilinear and the robot has to move along a rectilinear path, we proposed an optimal search strategy. Proposing a competitive search strategy, with minimum link for the simple robot, in more general classes of polygons is an attractive problem for future research.

References

1. Disser, Y., Ghosh, S. K., Mihalk, M., Widmayer, P.: Mapping a Polygon with Holes Using a Compass. Theoretical Computer Science, In Press, Corrected Proof, (Available online 18 December 2013)
2. Fekete, Sandor P., Joseph SB Mitchell, and Christiane Schmidt.: Minimum Covering with Travel Cost. Journal of Combinatorial Optimization, 24, 32-51 (2003)
3. Hammar, Mikael, Bengt J. Nilsson, and Mia Persson.:Competitive exploration of rectilinear polygons. Theoretical computer science 354.3, 367–378, (2006)
4. Icking, C., Klein, R., Langetepe, E.: An optimal competitive strategy for walking in streets. In STACS 99 (pp. 110–120). Springer Berlin Heidelberg (1999, January)
5. Katsev, Max, et al.: Mapping and pursuit-evasion strategies for a simple wall-following robot. Robotics, IEEE Transactions on 27.1, 113-128, (2011)
6. Klein, R.: Walking an unknown street with bounded detour. Computational Geometry, 1.6, 325–351 (1992)
7. Kumar Ghosh, Subir , and Sanjeev Saluja.: Optimal on-line algorithms for walking with minimum number of turns in unknown streets. Computational Geometry 8.5, 241-266 (1997)
8. Kumar Ghosh, Subir.: Computing the visibility polygon from a convex set and related problems. Journal of Algorithms 12.1, 75-95 (1991)
9. Lopez-Padilla, R., Murrieta-Cid, R., LaValle, S. M.: Optimal Gap Navigation for a Disc Robot. In Algorithmic Foundations of Robotics (pp. 123–138). Springer Berlin Heidelberg (2013)
10. Mitchell, J. S, Gnter Rote, and Gerhard Woeginger.: Minimum-link paths among obstacles in the plane. Algorithmica, 8.1-6, 431–459 (1998)
11. Sachs, S., LaValle, S. M., Rajko, S.: Visibility-based pursuit-evasion in an unknown planar environment. The International Journal of Robotics Research, 23.1, 3–26 (2004)

12. Suri, S., Vicari, E., Widmayer, P.: Simple robots with minimal sensing: From local visibility to global geometry. The International Journal of Robotics Research, 27.9, 1055–1067 (2008)
13. Tabatabaei, Azadeh, and Mohammad Ghodsi.: Walking in Streets with Minimal Sensing. In Combinatorial Optimization and Applications (pp. 361–372). Springer International Publishing (2013)
14. Tabatabaei, Azadeh, and Mohammad Ghodsi.: Walking in Streets with Minimal Sensing. Journal of Combinatorial Optimization http://link.springer.com/article/10.1007/s10878-014-9791-4 (2014)
15. Tabatabaei, Azadeh, and Mohammad Ghodsi.: Optimal Strategy for Walking in Streets with Minimum Number of Turns for a Simple Robot. In Combinatorial Optimization and Applications (pp. 101–112). Springer International Publishing (2014)
16. Tan, Xuehou, and Bo Jiang.: Minimization of the maximum distance between the two guards patrolling a polygonal region. Theoretical Computer Science, 532, 73–79 (2014)
17. Tovar, B., Murrieta-Cid, R., LaValle, S. M.: Distance-optimal navigation in an unknown environment without sensing distances. Robotics, IEEE Transactions on, 23.3, 506-518 (2007)
18. Tovar, B., La Valle, S. M., Murrieta, R.: Optimal navigation and object finding without geometric maps or localization. In Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on (Vol. 1, pp. 464–470). IEEE (2003, September)
19. Xu, Yinfeng, et al.: The canadian traveller problem and its competitive analysis. Journal of Combinatorial Optimization 18.2, 195-205 (2009).