Walking in Streets with Minimal Sensing

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 in an unknown street, starting from a point s, to reach a target t by a robot which has a minimal sensing capability. The goal is to decrease the traversed path as short as possible. The robot cannot infer any geometric properties of the environment such as coordinates, angles or distances. The robot is equipped with a sensor that can only detect the discontinuities in the depth information (gaps) and can locate the target point as soon as it enters in its visibility region. In addition, a pebble as an identifiable point is available to the robot to mark some position of the street. We offer a data structure similar to Gap Navigation Tree to maintain the essential sensed data to explore the street. We present an *online* strategy that guides such a robot to navigate the scene to reach the target, based only on what is sensed at each point and is saved in the data structure. Although the robot has a limited capability, we show that the detour from the shortest path can be restricted such that generated path by our strategy is at most 11 times as long as the shortest path to target.

1 Introduction

Path planning is one of the basic problems in computational geometry, online algorithms, and robotics [6, 8, 12]. Specifically, path planning appears in many applications where the environment is unknown and no geometric map of the scene is available [3]. In robot path planning, the robot's sensor is the only tool to collect information from the scene, and the volume of the information gathered from the environment depends on the capability of the sensor. A robot with a simple sensing model has many advantage such as: it is low cost, less sensitive to failure, robust against sensing uncertainty and noise, and applicable to many situations [3].

The robot that we use in this research, has a limited ability. It has an abstract sensor that can only detect the order of discontinuities in the depth information (or gaps) in its visibility region. Each discontinuity corresponds to a portion of the environment that is not visible to the robot, (Fig. 1). The robot assigns to every gap g a label L or R depending on which side of the gap the hidden

* This research was in part supported by a grant from IPM. (No. CS1390-2-01).

region is. Also, the robot recognizes a target point t when it is in the robot's omnidirectional and unbounded field of view. In order to cover the hidden region behind each gap, the robot moves towards the gap in an arbitrary steps. Note that the robot cannot measure any angles or distances to the walls of the scene or infer its position. In addition, we assume that the robot has access to a single pebble which is a detectable object that can be put anyplace and can be lifted again.

Throughout this paper, the workspace is assumed to be a restricted simple polygon called a street. A simple polygon P with two vertices s and t is called a street if the counter-clockwise polygonal chain R_{chain} from s to t and the clockwise chain L_{chain} from s to t are mutually weakly visible [7]. This means that each point on the left chain L_{chain} can see at least one point on the right chain R_{chain} and vice versa, (Fig. 1.a). In some literatures, a street is also known as L-R visible polygon [2]. A point robot that is equipped with the gap sensor starts navigating this environment from s to reach its target t. The robot has no geometric map of the scene and only based on the information gathered through the sensor has to make decisions to achieve the target.

Klein proposed the first competitive *online* strategy for searching a target point in a street [7]; called *walking in streets*. The robot employed in [7] is equipped with a 360 degree vision system. Also, it can measure each angle or distance to the walls of the street. As the robot moves, a partial map is constructed from what has been seen so far. Klein proved an upper bound of 5.72 for the competitive ratio (the ratio of the length of the traversed path to the shortest path from s to t) of this problem. Also, it was proved later that there is no strategy with the competitive ratio less than $\sqrt{2}$ for this problem. A strategy similar to Klein's with the competitive ratio of $\pi + 1$ has been introduced in [9, 10] which is robust under small navigation errors. Other researchers have presented several algorithms with the competitive ratios between $\sqrt{2}$ and the upper bound of 5.72 [8, 10]. Icking *et al.* presented an optimal strategy with the competitive ratio of $\sqrt{2}$ [6].

The limited sensing model that we use in this paper was first introduced by Lavalle *et al.* [16]. Gap Navigation Tree (GNT) has been proposed to maintain and update the gaps seen along the navigating path. This tree is built by detecting the discontinuities in the depth information and updated by the topological changes of the information. The topological changes are: appearances, disappearances, merges, and splits of gaps. Once the GNT is completed, it can encode the shortest path from its root (start point of the navigation) to any place in a simply connected environment. It is shown in [15] that, using this data structure, the globally optimal navigation is impossible in multiply connected environments, but locally optimal exploration can be achieved. Guilamo *et al.* [5, 13] presented an online algorithm for the well-known visibility problem pursuit-evasion in an unknown simply connected environment using GNT. As mentioned in [15], GNT is well suited for solving other visibility problems. An optimal search strategy using GNT is presented for a disc robot to find a target point t, starting from s in a simply connected environment [11].

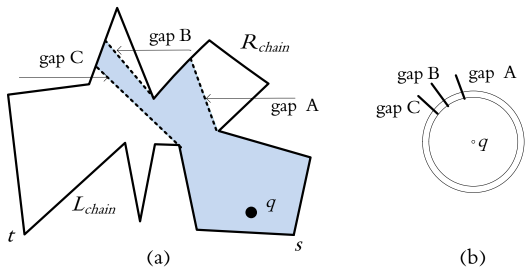

Fig. 1. (a) A street in which L_{chain} is the left chain and R_{chain} is the right chain. The colored region is the visibility polygon of the point robot q in the street. (b) The position of discontinuities in the depth information detected by the sensor.

Another minimal sensing model introduced by Suri and Vicari [14] for a simple robot. They assume that the robot can only sense the combinatorial (non-metric) properties of their surroundings. The sensor can detect vertices of the polygon in its visibility region, and can report if there is a polygon edge between consecutive vertices. The information maintain in two combinatorial vectors, called the combinatorial visibility vector (cvv) and the point identification vector (piv). Despite of minimal capability, they shown the robot can obtain many geometric reasoning and can accomplish many non-trivial tasks.

In this paper we propose an *online* search strategy for a point robot equipped with the gap sensor and the single pebble to reach the target point t in a street environment, starting from s. The minimal sensing model that we use here is in contrast with the strong sensing model that Klein used for walking in streets problem. A data structure that is maintained and updated similar to GNT is introduced for designing the robot search path. We show that the search path which is generated by our strategy is at most 11 times as long as the shortest path. Also, we show that if the robot has access to many pebbles, this ratio reduces to 9. To our knowledge, this is the first result providing some competitive ratio for walking in streets with the minimal sensing model.

2 GNT Data Structure and the Sensing Model

2.1 Gap Sensor

Gap sensor is a naive visual sensing model. At any position q of the environment, a cyclically ordered location of the depth discontinuities in the visibility region of the point $(V(q))$ is what the robot's sensor detects, as shown in Fig. 1. When the robot reports the discontinuities counterclockwise from a visibility region, it assigns a left label to a transition from far to near and assigns a right label to a transition from near to far [15]. The robot can only walk towards the gaps.

GNT data structure has been introduced as a mean to navigate in an unknown scene for the robot system. Here, we briefly explain the data structure from [15], and refer to it as T_g. The root of T_g is the robot's location. Each child of the root is a gap g that appears as the robot moves; these gaps are circularly ordered around the root. Each node, except the root, has a label of L or R. L means that the part of the scene which is hidden behind the gap is in the left side of the gap. R means that the part of the scene hidden behind the gap is in the right side of the gap, (Fig. 2).

As the robot moves, the critical events occur that change the combinatorial structure of the visibility region of the robot. There are four critical events in which T_g is updated: the appearance and disappearance events happen when the robot crosses the inflection rays, the merge and split events occur when the robot crosses the bitangent complements. In the disappearance event in which a gap g disappears, the node g will be eliminated from T_g. When a gap appears, a child is augmented to the root of T_g in a location that the circular ordering of the gaps is maintained. Each of these added nodes shows a portion of the environment that was so far visible, and now is invisible. These new nodes are specified as primitive (others are non-primitive). If a gap g splits into g_1 and g_2, then it will be replaced by the new nodes g_1 and g_2, (Fig. 2). If two gaps g_1 and g_2 merge into g , then g_1 and g_2, the adjacent children of the root, will be the children of a new node g which is added to the root.

The robot follows the non-primitive gaps until it reaches a point at which all leaf nodes are primitive. At this point, the robot has observed the entire environment. This data structure, after completion, can encode the shortest paths from the start point to any point of the environment.

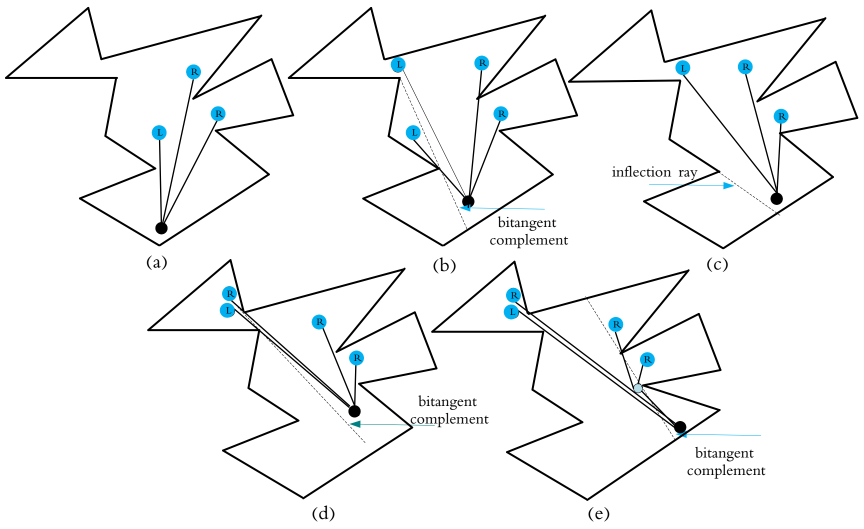

Fig. 2. The dark circle denotes the location of the robot. (a) Existing gaps at the beginning. (b) A split event. (c) A disappearance event. (d) Another split event. (e) A merge event.

2.2 The Sensor and Motion Primitive

All times, our robot's sensor reports the gaps, with their labels, in their counterclockwise cyclic order as they appear in its visibility region. The robot carries a pebble which is a marker device and is distinguishable for it. The robot can orient its heading with the gaps, and walks towards them in an arbitrary number of steps, for example: 2 steps towards a gap g_x, or 4 steps towards a gap g_y. Each step is a constant distance which is already specified for the robot by its manufacturer, for example it may be 1 meter, 2 meters and etcetera. When the robot moves towards a gap, it comes close to the gap, but the robot cannot report its distance to gaps and walls, size of gaps, and angles. Whenever a new event in the sensing of the environment happens, the robot can stop to make a reliable decision to reach the target. Also, the robot can move towards the pebble and the target, as soon as they enter in its visibility region, until it touches them.

3 Preliminarily Results

At each point p of the robot's search path, the gap sensor either sees the target, or achieves a set of gaps with the label of L or R (l-gap and r-gap for abbreviation). If the target is seen, the robot moves towards the goal and reaches it. In the other case in which the robot reports the position of the gaps (nonprimitive gaps), the robot should move towards the gaps to achieve the target.

Definition 1. *In the set of l-gaps, the gap which is in the right side of the others is called the most advanced left gap and is denoted by g_l. Analogously, in the set of r-gaps, the gap which is in the left side of the others is called the most advanced right gap and is denoted by g_r, (Fig. 3.a).*

Each gap is adjacent to a reflex vertex. The corresponding reflex vertices of g_l and g_r are denoted by v_l and v_r, (Fig. 3.a). The two gaps have the following property.

Lemma 1. *On any point of the robot search path, if the target is not visible, then it is behind one of the most advanced gaps.*

Proof. Let the target be behind of another gap, except g_l or g_r. Without loss of generality, it is behind an r-gap, so the points that are immediately behind g_r are not visible by any point on the opposite chain, this contradicts the definition of the street.

Above attribute of the two gaps is similar to the main feature of top most left packet and top most right pocket in [7].

As the robot moves in the environment, g_l and g_r may dynamically change. The critical events in which the structure of the robot's visibility region changes, can also change g_l and g_r. In the next section, we show how the critical events change the left most advanced gaps such that a sequence of the left most advanced gaps, $[g_{l1}, g_{l2}, ..., g_{lm}]$, appears in the robot's visibility region, while

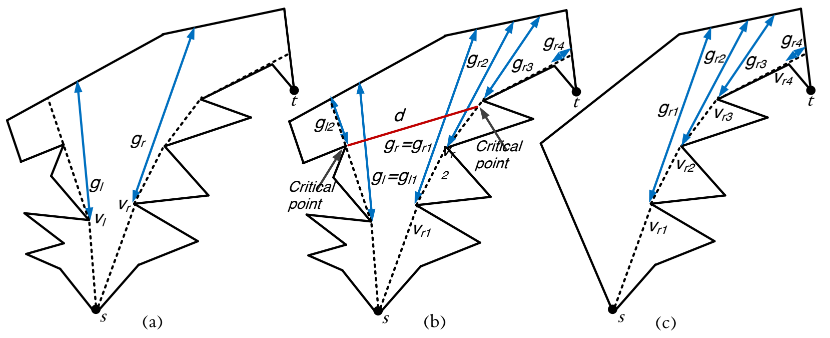

Fig. 3. (a) g_r and g_l are the most advanced gaps at the start point s. v_r and v_l are the corresponding reflex vertices. (b) Sequences of the most advanced gaps may occur, as the robot moves. The funnel situation which ends as soon as the robot crosses over the segment d. Dotted chains, starting from s, are the two convex chains of the funnel. (c) In this case there is only one most advanced gap, at start point s.

exploring the street. Similarly the sequence of the right most advanced gaps, $[g_{r1}, g_{r2}, ..., g_{rn}]$, may occur, (Fig. 3.b).

At each point, if there is exactly one of the two gaps (g_r or g_l), then the goal is hidden behind that gap. Thus, there is no ambiguity and the robot moves towards the gap, (Fig. 3.c). If both of g_r and g_l exist, then the target is hidden behind one of these gaps. This case is called a *funnel*, (Fig. 3.b). As soon as the robot enters a point in which both of g_r and g_l exist, a funnel situation starts. This case continues until one of g_r or g_l disappears, (Fig. 3.b), or they become collinear, (point 2 in Fig. 4.a). When the robot enters a point in which there is a funnel situation, the only non-trivial case in this navigation occurs.

In a funnel situation, previous strategies proposed by Kelin *et.al* [6–10] were based on choosing a walking direction within the angle between v_r and v_l. In other words, in this case, their robots select a point to move towards which is in equal distance with v_r and v_l and repeat this process until the funnel case ends. But, the robot that we use in this research cannot compute the point between v_r and v_l. So, applying their strategy for this robot is impossible. Before describing our strategy, we state some features of a street and the gaps that are applied in the algorithm.

When the robot enters in a funnel situation, there are two convex chains in front of it: *the left convex chain* that lies on the left chain (L_{chain}) of the street, and *the right convex chain* that lies on the right chain (R_{chain}) of the street, (Fig. 3.b). The two chains have the following main property.

Lemma 2. *When a funnel situation starts, shortest path from s to t lies completely on the left convex chain, or on the right convex chain of the funnel.*

Proof. Obviously, the point in which the funnel situation starts, belongs to shortest path from s to t. So, this claim is a straight result of the lemma 1 and the theorem below.

Theorem 1. *[4] For any vertex $v_j \in L_{chain}(or, v_j \in R_{chain})$, shortest path from s to v_j makes a left turn (respectively, a right turn) at every vertex of L_{chain} (respectively, R_{chain}) in the path.*

Definition 2. *Between the two convex chains, in a funnel situation, the one which is a part of the shortest path is called exact chain of the funnel.*

Lemma 3. *Each of the two convex chains, in a funnel situation, contains a point in which the funnel situation ends or a new funnel situation starts.*

Proof. While the robot explores the street in a funnel case, the situation ends in two conditions: (1) When the robot enters a point in which one of the most advanced gaps (g_l or g_r) disappears. The inflection ray of the gap intersects the two convex chains. The intersection points are the points which are claimed, (Fig. 3.b). (2) When the robot enters a point in which the two most advanced gaps are collinear. The bitangent of the corresponding reflex vertices of the current most advanced gaps intersects the two convex chains. So, the claimed points exist, (points 2 and 3 in Fig. 4.a).

We refer to the points, which is mentioned in the above lemma as *funnel critical points.* Clearly, one of the two points belongs to shortest path from s to t.

Lemma 4. *Assume the robot is walking along one of the convex chain of a funnel. The exact chain of the funnel can be specified as soon as the robot touches the critical point that belongs to the chain.*

Proof. There are two situations in which the robot touches a critical point: (1) The robot reaches a point in which one of the most advanced gaps disappears, obviously the convex chain which contains the existing gap is the exact chain, (Fig. 3.b). (2) The robot reaches a point in which g_l and g_r are collinear. If the point is the corresponding reflex vertex of the gap that the robot was moving towards, the chain that the robot was walking on it is the exact chain, (point 3 in Fig. 4.a). Otherwise the other chain is the exact chain, (point 2 in Fig. 4.a).

4 Main Strategy

Now, we explain our strategy for the robot to move in the street from s to t such that the generated path is at most a constant times as long as the shortest path. In the situation in which only one of the most advanced gaps exists each reasonable strategy directs the robot towards the gap.

The robot, based on the information gathered through its sensor and the pebble which it is equipped with, searches the scene. In the funnel situation, we lead the robot to reach the critical point. Our idea for directing the robot in this case is inspired by the algorithm for searching a point on a line, called doubling. In the doubling strategy, the robot moves back and forth on the line, such that at each stage i, it walks 2^i steps in one direction, comes back to the origin, walks 2^{i+1} steps in the opposite direction until the target is reached.

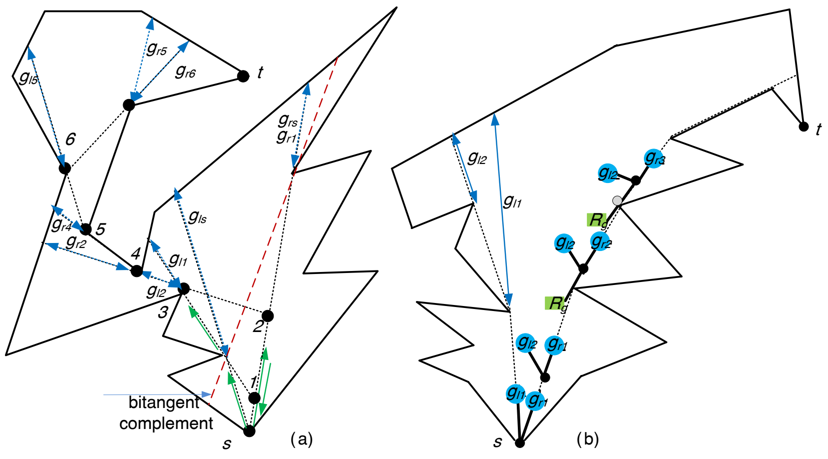

Fig. 4. (a)There is a funnel situation at start point s. Points 2 and 3 are the critical points of the funnel. g_{rs} and g_{ls} are the most advanced gaps at the start point. g_{ri} and g_{li} are the most advanced gaps at point i, for $i = 1, 2, ..., 6$. (b) Illustration of constructing and updating the data structure, as the robot walks along the right convex chain. Dark circle denote the robot's location and it is the root of the data structure. The path leads to R_g is the return path.

Theorem 2. *[1] The doubling strategy for searching a point on a line has a competitive factor of 9, and this is optimal.*

If we assume the two convex chains as a line then, by applying the doubling strategy on this line, we can find the critical point. Therefore, directing the robot along these two chains avoiding any detour to other places of the environment is what is important in this exploration.

Lemma 5. *The robot traces the left/ right convex chain and detects its critical point if and only it walks towards the left/ right most advanced gap maintaining the dynamically changes of the two most advanced gaps.*

Proof. When a funnel situation starts, the first vertex of the left/ right convex chain coincides with the corresponding reflex vertex of the currently left/ right most advanced gap. This most advanced gap doesn't change until the robot touches the reflex vertex. Then the first segment of the convex chain and robot's path are the same. Other segments are similarly coincident. As soon as the robot reaches a point in which one of the most advanced gaps disappears, or they are collinear, the critical point is achieved.

In the following subsection, we describe the process of constructing and updating the required data structure for leading the robot along the two convex chains and coming back to the origin (the point in which funnel case starts).

4.1 Data Structure

In the funnel situation, the robot puts a pebble on the point to mark this point as origin. In order to follow each of the two convex chains to reach the critical point, it must dynamically maintain the changes of g_r and g_l. Furthermore, the required information to come back to the origin must be preserved. This data is saved in a tree which we called S-GNT (street GNT).

The root of this tree is the start point of the funnel (current location of the robot). g_r and g_l are the only leaf of the tree, at the point. As the robot moves, the critical events, appearance, disappearance, merge and split, may dynamically change g_r and g_l. Moreover, these critical events generate the comeback path to origin as follows: (Assume the robot follows the right convex chain, in other words it moves towards g_r. The situation in which it moves towards g_l is symmetric and the S-GNT is constructed analogously.)

- When the robot crosses a bitangent complement of g_l and another l-gap, then g_l splits and will be replaced by the l-gap, (point 1 in Fig. 4.a).
- When the robot crosses a bitangent complement of g_l and an r-gap, then g_l splits into two gaps. g_r will be replaced by the r-gap. At this point g_l and g_r are collinear and the funnel situation ends, (point 2 in Fig. 4.a).
- When the robot crosses a bitangent of g_r and another r-gap, at the point in which g_r disappears, g_r will be replaced by the r-gap, in the tree. (disappearance and split events occur simultaneously.) In this situation, if there are more than one gap similar to the r-gap, g_r will be replaced by the one which is in the left side of the others, (point v_{r1} in Fig. 3.b).
- When the robot crosses a bitangent of g_r and another l-gap, at the point in which g_r disappears, g_l will be replaced by the l-gap, in the tree. (disappearance and split events occur simultaneously.) In this situation, if there are more than one gap similar to the l-gap, g_l will be replaced by the one which is in the right side of the others, (point 5 in Fig. 4.a).
- When the robot crosses over an inflection ray, each of g_l or g_r which is adjacent to the ray, disappears and is eliminated from the data structure. each of the critical point of the funnel in Fig. 3.b is an example for this event.
- When the robot crosses over an inflection ray, a gap may appear. If this gap hides the pebble that was so far visible, a child is augmented to the root of S-GNT in a location that the circular ordering of the gap and g_r and g_l is maintained. We refer to this gap as comeback gap. This gap is maintained in the tree for generating the comeback path to the origin, (point v_{r1} in Fig. 3.b). Other appearance events don't change the data structure.
- When the robot crosses a bitangent of reflex vertex of g_r and reflex vertex of the comeback gap, these two gaps merge. So, the comeback gap will be a child of a new node which is added to the root, (point v_{r2} in Fig. 3.b).

Note, the last two events update the data structure such that the return path to the origin is generated. In Fig. 4.b, the process of constructing the data structure, as the robot traces the right convex chain in the funnel situation in Fig. 3.b, is illustrated.

4.2 Algorithm

The robot starts navigating the environment based on the information gathered about the most advanced gaps until reaches a point in which the target is visible.

At each point, if there is exactly one of the two gaps (g_r or g_l), then the goal is hidden behind that gap. Thus, there is no ambiguity and the robot moves towards the gap.

In the funnel case in which both of g_r and g_l exist, the robot is not sure that the target is hidden behind which of these gaps. The robot put a pebble at this point, and saves the location of the two most advanced gaps at this point as g_{rf} and g_{lf}. At each stage i, the robot while constructing S-GNT, walks 2^i steps along the right convex chain, and returns to origin by following the return path, then walks 2^{i+1} steps along the opposite convex chain until a critical point of the funnel is achieved. As soon as the robot touches a critical point of current funnel, from lemma 4, the exact chain of the funnel is determined. So, at the critical point, the robot returns to the origin to pick up the pebble and walks along the exact chain to reach the target while constructing the S-GNT. The robot continues walking along the convex chain until the target is achieved or a new funnel case starts again. In the later case, the procedure for a funnel case (the doubling procedure) is repeated.

Note that at each stage i in a funnel case that the robot start going forth along one of the two convex chains, g_r and g_l in S-GNT are set to g_{rf} and g_{lf}, and as the robot moves S-GNT dynamically is constructed again, as explained in the previous section. Also, When no pebble is put on the environment, no return path generates in the S-GNT.

5 Correctness and Analysis

In this section, we show that the robot by following the path generated with our strategy achieves the target t starting from s. Also, we compare the length of the generated path with the shortest path and prove a constant competitive ratio for our strategy.

Theorem 3. *By executing our strategy, the robot rightly reaches target t, starting from start point s.*

Proof. In the walking in streets problem, the target constantly lies behind g_l or g_r. Thus, the events in which g_r and g_l are updated must be considered as critical events in the problem. Now, we show that these critical events are only the two types of the critical events: Split and disappearance. Each appearance event creates a primitive gap which was once visible by the robot. Obviously, the target is not behind this gap. A critical event which merges g_r and g_l occurs when the robot crosses the bitangent complement of the corresponding reflex vertices of the two gaps. As shown in Fig. 4.a, the bitangent complement either is in the left side of the line which connects the current position of the robot to g_l or is in the right side of the line which connects the current position of the robot to g_r. According to the algorithm, the robot cannot cross over the bitangent complement. Also, a most advanced gap merges with another gap at the point in which the most advanced gap will disappear. Hence, just the split and the disappearance are the critical events which change g_r and g_l. The merge and appearance are handled for constructing the return path to origin in data structure S-GNT.

We now compare the length of the path constructed by our *online* search strategy, and the length of shortest path. Each *online* walking strategy for a robot with the minimal sensing capability (the gap sensor) can significantly detour from the shortest path. Here, we prove a competitive ratio for the length of the generated search path by the algorithm.

Lemma 6. *In each funnel case, if we eliminate the robot movement to reach the critical point of the funnel, and comeback path to the origin of the funnel from the generated path by our strategy, the remained path and shortest path are coincide.*

Proof. When a funnel situation stars, the robot isn't sure which chain is the exact convex chain. From lemma 4, if the robot achieves the critical point, the exact chain is specified. So, the only detour from the shortest path is the movement to reach the critical point and comeback path to origin, in each funnel case.

Theorem 4. *By executing our strategy the robot can search a goal in an unknown street with a competitive ratio of at most 11. If the robot was allowed carrying many pebbles, it can search a goal with a competitive ratio of at most 9.*

Proof. By lemma 6, if an algorithm achieves a competitive factor in each funnel case, then it achieves the same ratio in every streets. So, we compare these two paths in one funnel in order to find the detour from the shortest path. The robot, using the information gather through its sensor about the most advanced gaps, searches the convex chains of the funnel, by executing the doubling strategy, until it reaches the critical point of the funnel. The robot traverses at most 9 times as long as the shortest path to reach the critical point. At this point, the robot comes back to the origin to pick up the pebble then walks along the exact chain. So, in each funnel situation the robot traverses at most 11 times as long as the length of shortest path to reach the critical point of the funnel which is on the shortest path.

6 Conclusions

In this study, we proposed an *online* strategy for the walking in streets problem for a point robot that has a minimal sensing capability. The robot can only detect the gaps and the target in the street. Also, it carry a pebble to mark some locations of the environment. Our strategy generates a path with a bounded detour from the shortest. We proved that our strategy has a competitive ratio of

11. Improving this upper bound can be considered as an opportunity for future research. Introducing more general classes of polygons which admit competitive searching with minimal sensing is an interesting open problem.

References

1. Baezayates, R. A., Culberson, J. C., Rawlins, G. J.: Searching in the plane. Information and Computation, 106(2), 234–252 (1993)
2. Das, G., Heffernan, P. J., Narasimhan, G.: LR-visibility in polygons. Computational Geometry, 7(1), 37–57 (1997)
3. Gfeller, B., Mihalk, M., Suri, S., Vicari, E., Widmayer, P.: Counting targets with mobile sensors in an unknown environment. In Algorithmic Aspects of Wireless Sensor Networks (pp. 32–45). Springer Berlin Heidelberg (2008)
4. Ghosh, S. K.: Visibility algorithms in the plane. Cambridge University Press (2007)
5. Guilamo, L., Tovar, B., LaValle, S. M.: Pursuit-evasion in an unknown environment using gap navigation trees. In Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on (Vol. 4, pp. 3456–3462). IEEE (2004, September).
6. 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).
7. Klein, R.: Walking an unknown street with bounded detour. Computational Geometry, 1(6), 325–351 (1992)
8. Kleinberg, J. M.: On-line search in a simple polygon. In Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms (pp. 8–15). Society for Industrial and Applied Mathematics (1994, January)
9. Lopez-Ortiz, A., Adviser-Ragde, P.: On-line target searching in bounded and unbounded domains. University of Waterloo (1996)
10. Lopez-Ortiz, A., Schuierer, S.: Simple, efficient and robust strategies to traverse streets. In Proc. 7th Canad. Conf. on Computational Geometry (1995)
11. Lopez-Padilla, R., Murrieta-Cid, R., LaValle, S. M.: Optimal Gap Navigation for a Disc Robot. In Algorithmic Foundations of Robotics X (pp. 123–138). Springer Berlin Heidelberg (2013)
12. Mitchell, J. S.: Geometric shortest paths and network optimization. Handbook of computational geometry, Elsevier Science Publishers B.V. North-Holland, Amsterdam (1998)
13. 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).
14. 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).
15. 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)
16. 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).