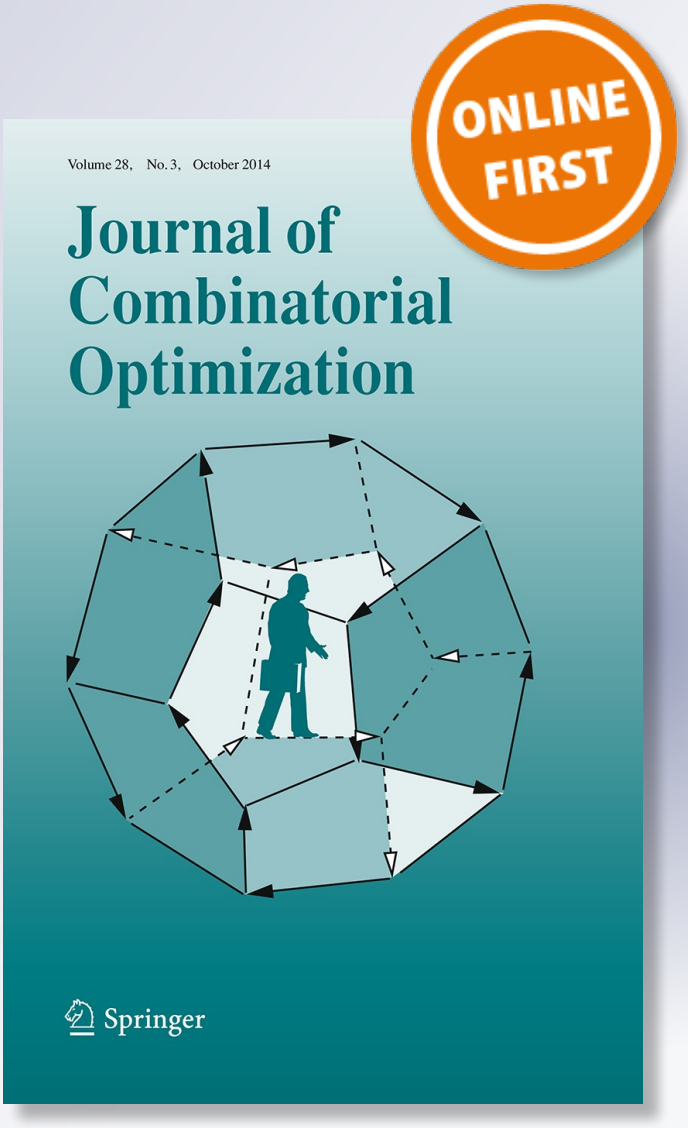
Walking in streets with minimal sensing

Azadeh Tabatabaei & Mohammad Ghodsi

Journal of Combinatorial Optimization

ISSN 1382-6905

J Comb Optim
DOI 10.1007/s10878-014-9791-4



Springer

J Comb Optim
DOI 10.1007/s10878-014-9791-4

Walking in streets with minimal sensing

Azadeh Tabatabaei · Mohammad Ghodsi

Abstract We consider the problem of walking a robot in an unknown polygon called "street", starting from a point s to reach a target t. The robot is assumed to have minimal sensing capability in a way that cannot infer any geometric properties of the environment, such as its coordinates, angles or distances; but it is equipped with a sensor that can only detect the discontinuities in the depth information (or gaps). Our robot can also locate the target point t as soon as it enters in robot's visibility region. In addition, one pebble is assumed to be available to the robot to be used as an identifiable point and to mark any position of the street. Our goal is to generate a shortest possible path for such robot from s to t in such a scene. We offer a data structure similar to Gap Navigation Tree to maintain the essential sensed data of the explored street. We present an *online* strategy that guides our robot to navigate the scene and reach the target. The strategy is based only on what is sensed at each point, and on what is saved in the data structure. Although the robot has a limited capability, we show that the robot's detour from the shortest path can be restricted such that our generated path is at most 11 times as long as the shortest path to the target. We also consider a special case of the problem in which the street is rectilinear and the search path has to be rectilinear. We propose a search strategy for this case that generates an L_1-shortest path from s to t.

A. Tabatabaei (✉)
Computer Engineering Department, Sharif University of Technology, Tehran, Iran
e-mail: atabatabaei@ce.sharif.edu

M. Ghodsi
Sharif University of Technology, Tehran, Iran
e-mail: ghodsi@sharif.edu

M. Ghodsi
School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran

Published online: 20 September 2014

Keywords Computational geometry · Robotics · Minimal sensing · Online algorithms · Competitive ratio

1 Introduction

Path planning is one of the basic problems in computational geometry, online algorithms, and robotics (Icking et al. 1999, Mitchell 1998, Tang et al. 2006, Xu 2009). In some robotics applications, the environment is unknown and no geometric map of the scene is available (Gfeller et al. 2008) in advance and one problem is to find a movement path for a robot to reach a target. In some path planning problems in robotics, the robot's sensors are the only tool available to collect information from the scene, and the volume of information gathered from the environment depends on the capability of the sensors. Among many variations, robots with a simple sensing model has attracted some researchers since the robot has low cost, is less sensitive to failure, is robust against sensing uncertainties and noise, and applicable to many situations (Chen et al. 2003, Gfeller et al. 2008).

In this research, we consider robots with limited abilities. More specifically, we assume that our robot 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 scene that is not visible to the robot (see Fig. 1). The robot assigns a label L or R to every gap g depending on which side of the gap the hidden region is located. Also, the robot recognizes a target point t whenever 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 number of steps. Note that the robot cannot measure any angles or distances to the walls of the scene, or infer its current position. In addition, we assume that the robot has access to a single pebble which is a detectable object that can be put any place and can be lifted again (Tabatabaei and Ghodsi 2013).

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 (Klein 1992). 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, (see Fig. 1a). In some literatures, a street is also known as L-R visible polygon (Das et al. 1997). A point robot equipped with the gap sensor starts navigating this environment from s to reach the target t. The robot has no geometric map of the scene and has to make decisions based only on the information gathered through the sensor to achieve the target.

Klein proposed the first competitive online strategy for searching a target point in a street (Klein 1992) and called this problem as *walking in streets*. The robot employed in Klein (1992) 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

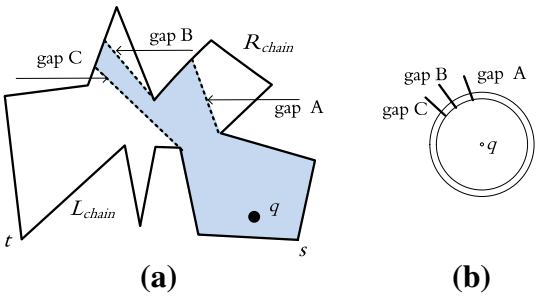


Fig. 1 **a** A street in which L_{chain} is the *left* chain and R_{chain} is the *right* chain. The colored region is the visibility polygon of the point robot q in the street. **b** The position of discontinuities in the depth information detected by the sensor

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 Lopez-Ortiz and Adviser-Ragde (1996), Lopez-Ortiz and Schuierer (1995) 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 (Kleinberg 1994, Lopez-Ortiz and Schuierer (1995)). Icking et al. presented an optimal strategy with the competitive ratio of $\sqrt{2}$ (Icking et al. 1999).

The limited sensing model applied in this paper was first introduced by Tovar et al. (2003). 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 (gaps) and updated by the topological changes of them. These 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 Tovar et al. (2007) 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. (2004), Sachs et al. (2004) presented an online algorithm for the well-known visibility problem pursuit-evasion in an unknown simply connected environment using GNT. As mentioned in Tovar et al. (2007), GNT is well suited for solving other visibility problems. An optimal search strategy for a disc robot, using GNT, is presented in Lopez-Padilla et al. (2013) to find a target point t, starting from s in a simply connected environment.

Another minimal sensing model was introduced by Suri et al. (2008) for a simple robot. They assumed that the robot can only sense the combinatorial (non-metric) properties of its 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. Two combinatorial vectors, combinatorial visibility vector (cvv) and point identification vector (piv), were proposed for maintaining the gathered data. Despite of the minimal capability, they showed that the robot can obtain many geometric reasoning and can accomplish many non-trivial tasks.

In this paper we present an online search strategy for a point robot equipped with the gap sensor, and a single pebble. The goal is to reach the target point t in a street environment, starting from s. The minimal sensing model that we apply for our robot 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

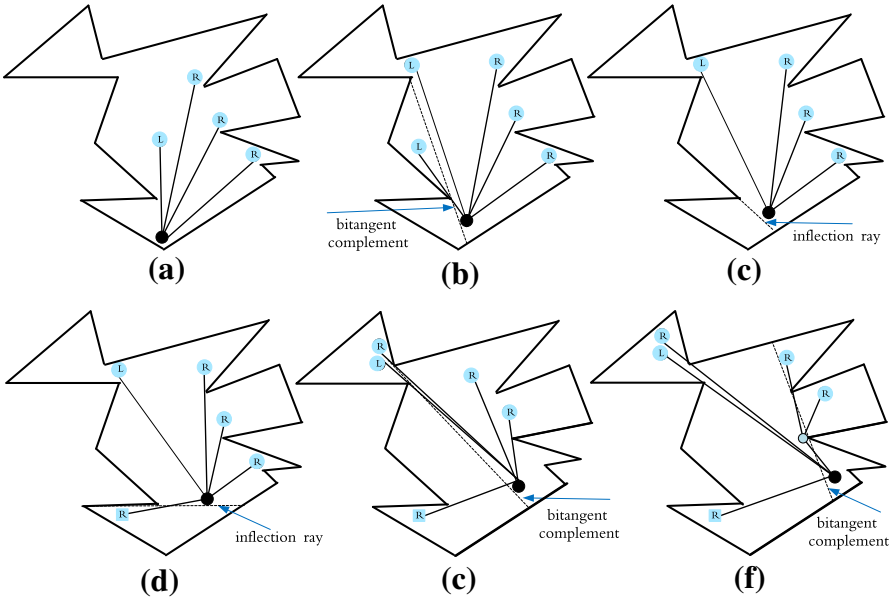


Fig. 2 The dynamically changes of the GNT when the robot moves towards a gap. The *dark circle* denotes the location of the robot, and *squares* and *other circles* display primitive and non-primitive nodes respectively

for designing the robot search path. We show that the search path which is generated by our strategy is at most 11 times longer than 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 that proposes a strategy with proven competitive ratio for walking in streets with the minimal sensing model.

2 The sensing model and motion primitive

2.1 Gap sensor

Gap sensor is a naive visual sensing model. At any position q of the environment, a cyclically ordered of discontinuities in the depth information in visibility region of the point (or $V(q)$) is what the robot's sensor detects. This is shown in Fig. 1b. Also, the robot assigns a left label to a transition from far to near and assigns a right label to a transition from near to far (Tovar et al. 2007). The robot can only walk towards the gaps.

GNT data structure has been introduced as a mean for navigating an unknown scene for the robot system. Here, we briefly explain the data structure from Tovar et al. (2007), 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 in the robot's visibility region; 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. 2a).

As the robot moves, several critical events may occur that can change the combinatorial structure of the robot visibility region. There are four types of such critical events each of which will update T_g. These are: the appearance and disappearance events that happen when the robot crosses the inflection rays, the merge and split events that occur when the robot crosses the bitangent complements. When a disappearance event occurs, a gap g disappears and the corresponding node to the gap g is eliminated from T_g (see Fig. 2c). When a gap appears, a child is augmented to the root of T_g in a location where the circular ordering of the gaps is maintained. Each of the added nodes shows a portion of the environment that was so far visible, and now is invisible to the robot. These new nodes are specified as primitive (see Fig. 2d). 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, (shown in Fig. 2b, e). 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 (see Fig. 2f). 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.

2.2 Motion primitive

A point robot starts navigating the street with unit speed starting from the fixed point s. Its sensor reports the gaps and their labels in their counterclockwise cyclic order as they appear in robot's visibility region. The robot carries a pebble which is a marker device and is distinguishable. The robot can orient its heading to the gaps and walks towards them in an arbitrary number of steps; for instance: two steps towards a gap g_x, or four steps towards a gap g_y. Each step is a constant distance which is already specified for the robot by its manufacturer. The robot moves towards a gap, but it cannot report its distance to the gaps or the walls, nor it can measure the size of gaps, and their angles. Whenever a critical event occurs, the robot can stop to make a reliable decision to reach the target. Also, the robot moves towards the pebble and the target t as soon it sees them and continues its movement until it touches the pebble or the target.

3 Preliminarily results

At each point p of the robot's search path, its 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 visible, the robot moves towards the target to reach it. When the robot reports the position of the gaps (non-primitive gaps), it should move towards the gaps to find the target.

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

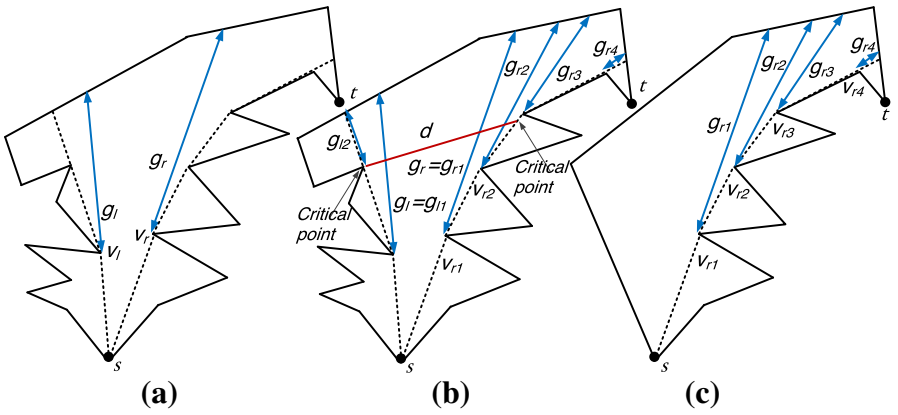


Fig. 3 **a** g_r and g_l are the most-advanced gaps at the start point s. v_r and v_l are the corresponding reflex vertices. **b** Sequences of the most-advanced gaps may occur, as the robot moves. The funnel situation which ends as soon as the robot crosses over the segment d. *Dotted* chains, starting 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

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, (see Fig. 3a). The following property holds:

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

Proof Let the target be behind a gap other then g_l or g_r. Without loss of generality, assume that 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.

The above property of the two gaps is similar to the main feature of the top-most left packet and top-most right pocket in Klein (1992). As the robot moves in the environment, g_l and g_r may dynamically change. The critical events that change the structure of the robot's visibility region 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, $[g_{l1}, g_{l2}, \ldots, g_{lm}]$, appears in the robot's visibility region, while exploring the street. Similarly the sequence of the right most-advanced gaps, $[g_{r1}, g_{r2}, \ldots, g_{rn}]$, may occur (see Fig. 3b).

At each point, if there is exactly one of the two gaps (g_r or g_l), then the target is hidden behind the gap. Thus, there is no ambiguity and the robot moves towards the gap, (Fig. 3c). 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* (see Fig. 3b). 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 (see Fig. 3b), or they become collinear, (point 2 in Fig. 4a). When the robot enters a point in which there is a funnel situation, the only non-trivial case in this navigation occurs.

In earlier papers, when such a funnel situation occurs, Kelin et.al proposed in Icking et al. (1999), Klein (1992), Kleinberg (1994), Lopez-Ortiz and Adviser-Ragde (1996), Lopez-Ortiz and Schuierer (1995) to walk along a direction between g_r and g_l. In other

words, in this case, the robot selects a point to move towards which is in equal distance with v_r and v_l and repeats 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 is impossible for us. Before describing our strategy, we state some features of a street and the gaps applied in our 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 (see Fig. 3b). 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 *Ghosh (2007) For any vertex $v_j \in L_{chain}$(or, $v_j \in R_{chain}$), the 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 When the robot explores the street in a funnel case, the situation ends as the robot enters a point in which either one of the most-advanced gaps (g_l or g_r) disappears or the two gaps become collinear. In the former case, the inflection ray of the disappeared gap intersects the two convex chains. The intersection points are the points which are claimed (see Fig. 3b). In the later case, a new funnel situation starts; the most-advanced gaps of this funnel are collinear at the point. 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. 4a).

We refer to the points 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 reaches the critical point of the chain.*

Proof There are two situations in which the robot reaches 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. 3b). (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. 4a). Otherwise the other chain is the exact chain, (point 2 in Fig. 4a).

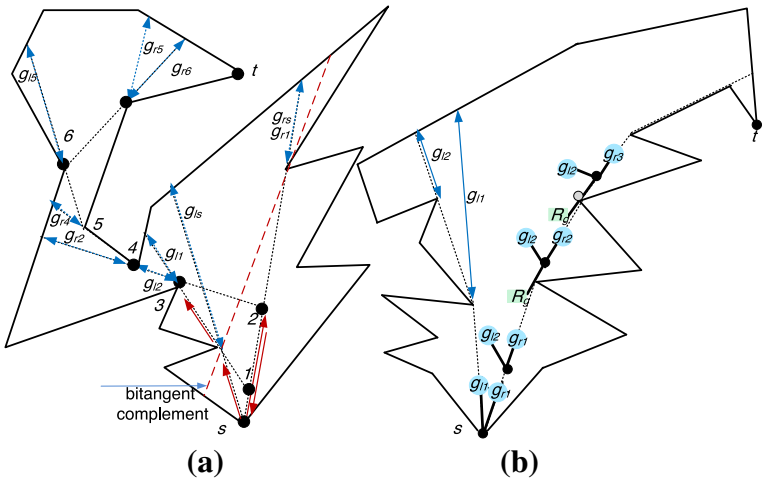


Fig. 4 **a** There is a funnel situation at start point s. Points 2 and 3 are the critical points of the funnel. g_{rs} and g_{ls} are the most-advanced gaps at the start point. g_{ri} and g_{li} are the most-advanced gaps at point i, for $i = 1, 2, \ldots, 6$. **b** Illustration of constructing and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

4 Main strategy

Now, we explain our strategy for searching the street from s to t such that the generated path is at most a constant times longer than 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. In the funnel situation, we lead the robot to reach a critical point of the funnel. 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.

Theorem 2 *Baezayates et al. (1993) 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 of a funnel as a line, by applying the doubling strategy, we can find a critical point of the funnel; the closer point to the origin. Note that the other critical point of the funnel may enormously be far from the origin. Therefore, directing the robot along these two chains avoiding any detour to other places, for executing the doubling strategy, is an important step of our algorithm.

Lemma 5 *The robot traces the left/right convex chain and detects its critical point if and only if it walks towards the left/right most-advanced gap maintaining the dynamic 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 reflex vertex of the current left/right most-advanced gap. The most-advanced gap that the robot is walking towards does not change until the robot touches its reflex vertex, or the funnel situation ends (the two gaps become collinear). 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.

While the robot moves towards a most-advanced gap, the positions of the two gaps dynamically change. In the next section, the events that update the gaps are explained. Furthermore, a data structure for maintaining the requirement data to return to origin in the funnel situation is introduced.

4.1 Data structure

As the robot moves along each of the two convex chains, the critical events, appearance, disappearance, merge, and split, may dynamically change g_r and g_l as follows: (Assume the robot moves towards g_r. The situation in which the robot moves towards g_l is symmetric, and updates the gaps analogously.)

1. 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. 4a).
2. 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. 4a).
3. 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. (disappearance and split events occur simultaneously.) In this situation, if there are more than one gap, same as the r-gap, g_r will be replaced by the one in the left side of the others (point v_{r1} in Fig. 3b).
4. 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. (disappearance and split events occur simultaneously.) In this situation, if there are more than one gap, same as the l-gap, g_l will be replaced by the one in the right side of the others (point 5 in Fig. 4a).
5. When the robot crosses over an inflection ray, each of g_l or g_r adjacent to the ray, disappears and is eliminated. Each of the critical point of the funnel in Fig. 3b is an example for this event.

In Lemma 6, we show that these events are the ones which merely change the position of g_l and g_r.

Lemma 6 *The critical events which update g_r and g_l are only of the two types of the critical events: split and disappearance.*

Proof 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 or g_l with another gap, occurs when the robot crosses the bitangent complement of

their corresponding reflex vertices of the two gaps. In order that such an event occurs, the bitangent complement should be either in the left side of the line which connects the current position of the robot to g_l or in the right side of the line which connects the current position of the robot to g_r. Due to our strategy, walking towards the most-advanced gaps, the robot cannot cross over the bitangent complement. So, these gaps do not merge with another gap at all (see Fig. 4a).

In the funnel situation, the robot puts a pebble on the point to mark it as the origin, and starts walking along one of the chains to reach the critical point. In order to execute the doubling strategy, the data needed for returning to the origin should be memorized. This data is saved in a new tree called S-GNT (street GNT). Initially, when the robot is placed in the point, robot's location is the root node, and g_l and g_r are the set of the children of S-GNT. The root node moves along with the robot, and g_l and g_r, as already explained, will be updated. The returning path to the origin is constructed as follows: (Assume the robot walks along the right convex chain. The situation in which the robot moves along the other is symmetric, and the data structure is constructed analogously.)

- 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 a *comeback gap*. This gap is maintained in the tree for generating the comeback path to the origin (shown as the first reflex vertex of the right convex chain in Fig. 4b). Other appearance events do not 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 added to the root, (the second reflex vertex of the right convex chain in Fig. 4b). The new node is a node of comeback path.

In Fig. 4b, the process of constructing the data structure, as the robot traces the right convex chain, is illustrated.

4.2 Algorithm

The robot starts navigating the environment based on the information gathered about the most-advanced gaps until the target is reached.

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

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 puts a pebble at this point, and moves back and forth along the two convex chain such that at each stage i, the robot (1) goes 2^i steps along one of the two convex chains while constructing S-GNT, then it returns back to the origin using S-GNT, and (2) goes 2^{i+1} steps along the opposite convex chain while constructing S-GNT, then it returns to the origin using S-GNT. At each funnel situation, the robot continues executing the doubling strategy until the critical point of the funnel is achieved. As soon as the robot touches a critical point of

the 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 maintaining and updating g_r and g_l. The robot continues walking along the chain until the target is achieved or a new funnel case starts again. In the later case, the procedure of the funnel case (the doubling procedure) is repeated.

Note that at each stage i, in a funnel case that the robot starts going forwards along one of the two convex chains, a new S-GNT is dynamically constructed. The robot goes back using the S-GNT.

5 Correctness and analysis

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

Theorem 3 *The strategy of searching the street, starting from s, terminates while the target t is achieved.*

Proof The key observation in our strategy is that the target is constantly hidden behind the most-advanced gaps. Even though moving towards a most-advanced gaps may reveal new gaps, or may update the two gaps via splitting, the number of such events is finite because each corresponds to a crossing of a bitangent complement. When the strategy forces a most-advanced gap to disappear, the robot is one step closer to the target. So, the procedure must terminate with the target being achieved.

We now compare the length of the path constructed by our search strategy, and the length of the shortest path and prove its competitive ratio.

Lemma 7 *If we eliminate the robot movement to reach the critical point, and the comeback path to the origin to pick up the pebble in each funnel situation, then the remaining path coincides with the shortest path.*

Proof When a funnel situation starts, it is not certain for the robot to know which chain is the exact convex chain. From Lemma 4, if the robot achieves the critical point, the exact chain is specified. So for each funnel case, the only detour from the shortest path is the back and forth movement to reach the critical point.

Theorem 4 *Our strategy for searching an unknown street achieves a competitive ratio of 11. Allowing the robot to carry many pebbles reduces the competitive ratio to 9.*

Proof By Lemma 7, if an algorithm achieves a competitive factor in a funnel case, then it achieves the same ratio in every streets. So, we consider the generated path in one funnel in order to find amount of deviation from the shortest path. In each funnel situation, the robot using the S-GNT executes the doubling strategy on the two convex chain, until it reaches the critical point of the funnel. The robot traverses at most 9 times longer than the shortest path to reach the critical point. At this point, the robot

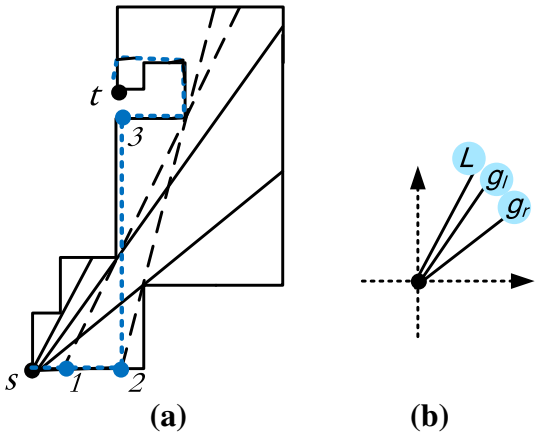


Fig. 5 **a** A rectilinear street. The dotted path is the generated path by our strategy. **b** The data structure at the start point

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 longer than the shortest path to reach the critical point of the funnel. The point is on the shortest path.

6 Walking in rectilinear streets with rectilinear path

In this section, we present an algorithm for leading the robot in a rectilinear street to move from s to t such that the traversed path is as short as possible, and the robot follows a rectilinear path.

Without loss of generality, assume that the edges of the polygon are vertical or horizontal. In other words, they are parallel to X or Y axis. So, the segments of a rectilinear path are parallel to X or Y axis. The robot, while detecting the gaps, moves vertically or horizontally (along positive and negative direction of the axis) (see Fig. 5a). Throughout the street navigation, we maintain the obtained information from the gap sensor in a data structure, that dynamically changes, to construct an online search strategy. This data will be represented in a tree with depth of one. At the start point, the root of the tree is current position of the robot, and children of the root are the sensed gaps maintained in circular order. Root of the tree, same as S-GNT, moves along with the robot. Also, the critical events update its children (see Fig. 5b).

At any point during the searching, there are two cases: either there are some gaps which hide the target t or the target is visible. We first discuss below the strategy of the algorithm when t is not visible. From Lemma 1, it should be hidden behind one of the most-advanced gaps. The possible location of the two most-advanced gaps are as follows:

1. The two gaps lie in the same quadrant of the X-Y plane, or one of two gaps only exists (Ghosh and Saluja 1997). If there is a rectilinear most-advanced gap (the gap lies on the X or Y axis), then the robot moves towards the rectilinear gap, (Fig. 6). Otherwise, the robot takes one of the rectilinear direction of the quadrant and moves along it.

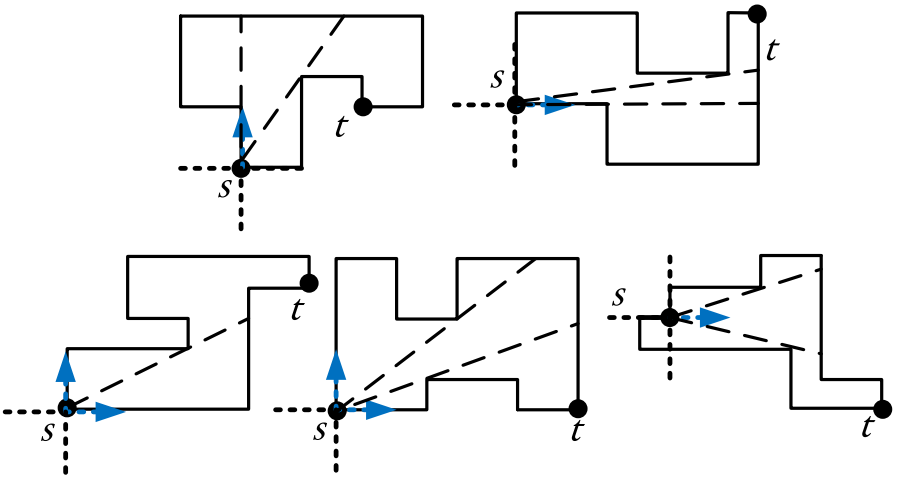


Fig. 6 Possible locations of the existing gaps in the data structure at the start point. At each case, the *arrow(s)* shows the direction that the robot may move along it, by our strategy

2. The two gaps lie in the two consecutive quadrant (Ghosh and Saluja 1997). In this situation, the robot takes the rectilinear direction which is between the two gaps, (Fig. 6).

We now characterize our strategy for turning while the robot is walking along any direction. During the moving, due to the critical events, the position of the gaps appearing to the robot dynamically change. Split and disappearance events update g_l and g_r as explained in Sect. 4.1, (point 1 in Fig. 5a). Once updated, the tree data structure changes and the robot may make a turn based on which of the above two conditions occurs. Remarkably, an appearance event generates a gap that hides a portion of the street, which was already explored. Such event does not add any node to the data structure. In contrast with our strategy for walking in general street, by our strategy for leading the robot in rectilinear street, a merge event may occur. So, a most-advanced gap merges with one of the existing gap (see point 2 in Fig. 5a). Once the merge event occurs, the robot makes a turn and moves along the other axis containing the most-advanced gaps. During the moving, as the robot hits a wall, it decides to turn based on the current position of the gaps in the tree (see point 3 in Fig. 5a).

Using the above strategy, the robot searches the street until it achieves a point in which the target is visible. The robot continues walking along the current direction until: (i) t is visible along the perpendicular direction to the current direction (the other axis of the quadrant), or (ii) t becomes invisible. At each of these cases, the robot turns along the perpendicular direction to the current direction. However this process is repeated until the target t is achieved (Fig. 5).

An argument, similar to the one for Theorem 3, proves the theorem below.

Theorem 5 *Our strategy for searching the rectilinear street terminates with the target achieved.*

Theorem 6 *The search path generated by our strategy to reach the target t, starting from s, is an L_1-shortest path. Also, the strategy achieves a competitive ratio of $\sqrt{2}$ in the L_2-metric.*

Proof During the movement, due to the location of the most-advanced gaps, the robot selects a vertical or a horizontal direction. Assume that (x_l, y_l) and (x_r, y_r) are the reflex vertices of g_l and g_r, respectively, and (x_s, y_s) is the current location of the robot. If the robot selects a horizontal direction then it keeps the direction until a point p is achieved such that $|x_s - x_p| = |x_s - x_l|$, or $|x_s - x_p| = |x_s - x_r|$. Similarly, if the robot selects a vertical direction then it keeps the direction until a point p is achieved such that $|y_s - y_p| = |y_s - y_l|$, or $|y_s - y_p| = |y_s - y_r|$. Because at the point a critical event occurs. Note that the robot sometimes changes its direction before reaching the point p; when a merge event arises (shown in Fig. 5a). Every path from s to cover the hidden region behind each of the gaps expands in X-direction or Y-direction at least as much as our path does. This shows that the point p lies on each L_1-shortest path from s to t. Since each L_1-shortest path is at most $\sqrt{2}$ times as long as the corresponding L_2-shortest path, competitive ratio of the algorithm is $\sqrt{2}$.

7 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 carries a pebble to mark some locations of the environment. Our strategy generates a path with a bounded detour from the shortest path. We proved that our strategy has a competitive ratio of 11. A special case in which the robot is forced to move along a rectilinear path in a retailer street is considered. We proposed an algorithm for the problem which generates an L_1-shortest path from s to t. Introducing more general classes of polygons which admit a competitive searching with a minimal sensing model is an interesting open problem. Improving the competitive ratio can be considered as an opportunity for the future research.

References

Baezayates RA, Culberson JC, Rawlins GJ (1993) Searching in the plane. Inform Comput 106(2):234–252

Chen DZ, Hu XS, Xu J (2003) Computing optimal beams in two and three dimensions. J Combin Optim 7(2):111–136

Das G, Heffernan PJ, Narasimhan G (1997) LR-visibility in polygons. Comput Geom 7(1):37–57

Gfeller B, Mihalk M, Suri S, Vicari E, Widmayer P (2008) Counting targets with mobile sensors in an unknown environment. Algorithmic aspects of wireless sensor networks, pp 32–45. Springer, Berlin

Ghosh SK (2007) Visibility algorithms in the plane. Cambridge University Press, Cambridge

Ghosh SK, Saluja S (1997) Optimal on-line algorithms for walking with minimum number of turns in unknown streets. Comput Geom 8(5):241–266

Guilamo L, Tovar B, LaValle SM (2004) Pursuit-evasion in an unknown environment using gap navigation trees. In Intelligent robot's and systems proceedings vol. 4, pp 3456–3462

Icking C, Klein R, Langetepe E (1999) An optimal competitive strategy for walking in streets. In STACS 99, pp 110–120. Springer, Berlin

Klein R (1992) Walking an unknown street with bounded detour. Comput Geom 1(6):325–351

Kleinberg JM (1994) On-line search in a simple polygon. In Proceedings of the fifth annual ACM-SIAM symposium on discrete algorithms, pp 8–15

Lopez-Ortiz A, Adviser-Ragde P (1996) On-line target searching in bounded and unbounded domains. University of Waterloo, Waterloo

Lopez-Ortiz A, Schuierer S (1995) Simple, efficient and robust strategies to traverse streets. In: Proceedings of th 7th Canadian conference on computational geometry

Lopez-Padilla R, Murrieta-Cid R, LaValle SM (2013) Optimal gap navigation for a disc robot. In Algorithmic foundations of robotics, pp 123–138. Springer Berlin

Mitchell JS (1998) Geometric shortest paths and network optimization. Handbook of computational geometry. Elsevier Science Publishers B.V. North-Holland, Amsterdam

Sachs S, LaValle SM, Rajko S (2004) Visibility-based pursuit-evasion in an unknown planar environment. Int J Robotics Res 23(1):3–26

Suri S, Vicari E, Widmayer P (2008) Simple robots with minimal sensing: from local visibility to global geometry. Int J Robotics Res 27(9):1055–1067

Tabatabaei A, Ghodsi M (2013) Walking in streets with minimal sensing. In Combinatorial optimization and applications, pp 361–372. Springer, Berlin

Tang J, Xue G, Zhang W (2006) Reliable ad hoc routing based on mobility prediction. J Combin Optim 11(1):71–85

Tovar B, Murrieta-Cid R, LaValle SM (2007) Distance-optimal navigation in an unknown environment without sensing distances. Robotics IEEE Trans 23(3):506–518

Tovar B, La Valle SM, Murrieta R (2003) Optimal navigation and object finding without geometric maps or localization. In Robotics and automation vol 1, pp 464–470

Xu Y et al (2009) The canadian traveller problem and its competitive analysis. J Combin Optim 18(2):195–205