An Optimal Solution to Room Search Problem

Binay Bhattacharya* John Z. Zhang† Qiaosheng Shi* Tsunehiko Kameda*

Abstract

A room is a simple polygon with a prespecified point, called the door, on its boundary. A search starts at the door and must detect any intruder in the room and guarantee that no intruder escapes through the door. Depending on where the door is placed, such a room may or may not be searchable. We consider the two-guard search model, where the two guards are required to move on the boundary of a room and be always mutually visible during a search. By making extensive use of the so-called visibility diagram, we present a simple characterization of searchable rooms by two guards and propose a linear algorithm to check it.

1 Introduction

The *room search* problem is one variation of the polygon search problem [7]. Let $P(d)$ denote a *room*, which is a simple polygon P with a designated point d on its boundary, called a *door*. We consider two guards in our work, which are required to move on a room's boundary and be always mutually visible. Park *et al.* [6] investigated a similar problem.

Our characterization of a searchable room is essentially the same as the one in [6]. However, the correctness proof of the characterization in [6] is not easy to follow and the corresponding searchability checking requires $O(nlogn)$ time. In contrast, our characterization is much simpler in terms of its correctness proof, representation and interpretability. In addition, we propose a linear-time algorithm for checking a room's searchability. We attribute all these to the visibility diagram [4] that grasps the visibility information in a given room.

The rest of the paper is organized as follows. In Section 2, we discuss the notation and the visibility diagrams. We apply them to the room search problem in Section 3. We present our linear algorithm for checking the searchability of a room in Section 4. Finally, we summarize our work in Section 5.

2 Preliminaries

A (simple) polygon P consists of n *vertices* ($n \geq 3$) and n *edges* connecting adjacent vertices. The *boundary* of P, denoted ∂P, consists of all its edges and vertices. We designate that $\partial P \subseteq P$. The vertices immediately preceding and succeeding vertex v clockwise are denoted by $Pred(v)$ and $Succ(v)$, respectively. For any two points $a, b \in \partial P$, the open (closed, resp.) polygonal chain ∂P from a to b clockwise is denoted by $\partial P_{cw}(a, b)$ ($\partial P_{cw}[a, b]$, resp.). Given a prespecified point $d \in \partial P$ (called *door*), for two points $a, b \in \partial P$, if moving clockwise from d, a is encountered before b, we write $a \prec_d b$. For all $a \in \partial P$ with $a \neq d$, we write $d \prec_d a$.

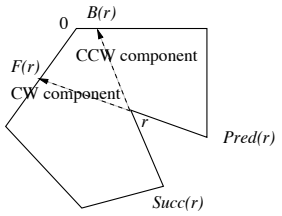


Figure 1: Illustration for some terms.

A vertex whose interior angle between its two incident edges is more than 180^o is called a *reflex vertex*. Consider reflex vertex r in Fig. 1. Extend the edge $(Succ(r), r)$ towards the interior of P, and let $B(r) \in \partial P$ denote the *backward extension point*, where this extension leaves P for the first time. We call the polygonal area formed by $\partial P_{cw}[r, B(r)]$ and the chord $\overline{rB(r)}$ the *clockwise component* associated with r, denoted as $C_{cw}(r)$. Similarly, the extension of $(Pred(r), r)$ determines *forward extension point*, $F(r)$, and the *counter-clockwise component* associated with r, denoted as $C_{ccw}(r)$, is bounded by $\partial P_{cw}[F(r), r]$ and the chord $\overline{rF(r)}$. A clockwise (counter-clockwise, resp.) component is *redundant* if it is a superset of another clockwise (counter-clockwise, resp.) component.

Two points u and v are said to be *mutually visible* if the line segment $\overline{uv}$ is completely contained inside P[1].

We pick a point on ∂P as the origin, and measure all distances along ∂P clockwise from it. Let $|\partial P|$ denote the length of ∂P. For $x \in \mathbf{R}$,[2] x represents the point on ∂P which is at distance $x - k|\partial P|$ from the origin, where k is an integer such that $0 \leq x - k|\partial P| < |\partial P|$.

Let $x, y \in \mathbf{R}$. The *visibility space* (*V-space* for short), denoted by $\mathcal{V}$, consists of the infinite area between and including the lines $y = x$ (the *start line* S) and $y = x - |\partial P|$ (the *goal line* G), as shown in Fig. 2 [4]. Therefore,

*School of Computing Science, Simon Fraser University, {binay, qshi1, tiko}@cs.sfu.ca

†Department of Math and Computer Science, University of Lethbridge, zhang@cs.uleth.ca

[1]The definition of mutual visibility in [5] is different from ours adopted here.

[2]$\mathbf{R}$ denotes the set of all real numbers.

we have $(x, y) \in \mathcal{V}$ if and only if $x - |\partial P| \leq y \leq x$.

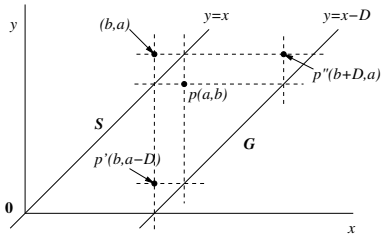


Figure 2: Visibility space ($D = |\partial P|$).

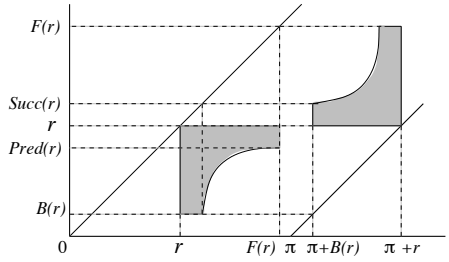


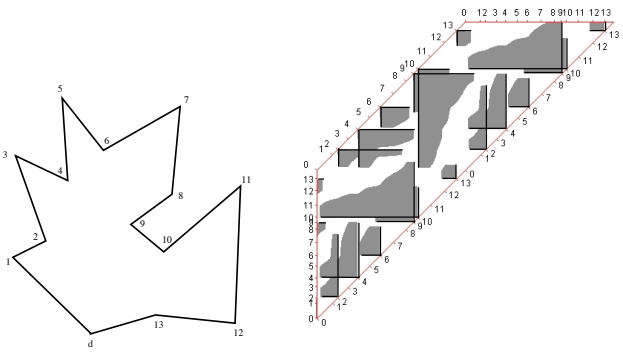
Figure 3: A section of the V-diagram for the polygon in Fig. 1 ($\pi = |\partial P|$).

The *visibility diagram* (*V-diagram* for short) for a given polygon is drawn in the V-space by making some areas in it gray as follows: point $(x, y) \in \mathcal{V}$ is gray if points x and y are mutually invisible. Our V-diagram contains essentially the same information as the X-diagram [5], which grasps the visibility information between any two boundary points for a given polygon, but has the advantage of not wrapping around when we make a complete circle around the boundary. For example, Fig. 3 is the V-diagram for the polygon in Fig. 1. In another example shown in Fig. 4, we highlight the parts of the boundaries of gray regions, which are the maximal line segments touching either line S or G, as shown in Fig. 5 (a). We call the set of these line segments the *skeleton* of the V-diagram. It is known that the V-diagram and its skeleton are topologically equivalent [4]. Therefore, we will focus on the skeletal V-diagram (*SV-diagram*, for short) in the sequel.

3 Searching a Room by Two Guards

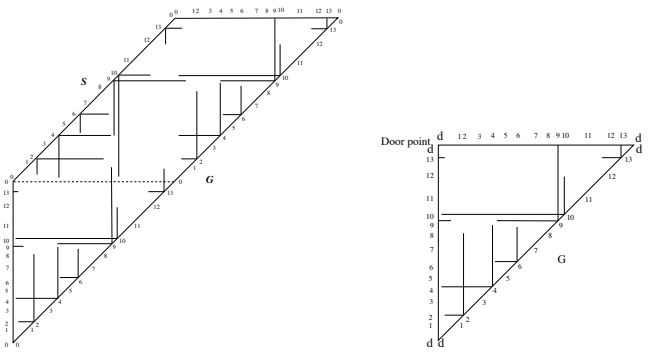
Given a room $P(d)$, two guards start their search from d and move on the boundary in the opposite direction. During the search, the two guards are always mutually visible and the intruders cannot escape from the room through d. If the two guards eventually meet at some same point on the boundary, we say that the room is *searchable*.

We use a coordinate point (x, y) in the V-space to represent the current positions of the two guards on ∂P, with x for the left guard and y for the right guard, viewed from d. Since no guard can go across the door



(a) (b)

Figure 4: (a) An example polygon; (b) Its V-diagram.



(a) (b)

Figure 5: (a) SV-diagram corresponding to Fig. 4 (b); (b) Relevant part for room search.

at any time, this coordinate point is within the area bounded by $x \geq d$ and $y \leq d$. For example, for the polygon in Fig. 4 (a), if $d = 0$, we need only consider the triangular part of the corresponding SV-diagram that lies below the dashed line, as shown in Fig. 5 (b).

The initial positions of two guards (d, d) are called the *door point* in the SV-diagram. Our analysis is based on the following important fact [4].

Proposition 1 *A room is searchable by two guards if and only if there exists a path from (d, d) to line G without crossing any skeletal segments in its SV-diagram bounded by $x \geq d$ and $y \leq d$.* ■

Call a maximum white area in the SV-diagram a *cell*. It is clear that if a cell in the SV-diagram does not touch both lines S and G then no search path can go through the cell. We call such a cell a *trap cell*. In the following figures, we will draw a room as a circle and mark only the relevant reflex vertices on its boundary. We list in Fig. 6 some obvious patterns of reflex vertices that put the door point on the boundary of a trap cell in the SV-diagram. Note that Fig. 6 does not exhaust

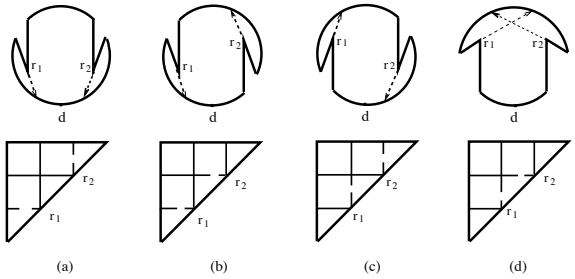


Figure 6: The reflex vertex patterns that cause trap cells touching the door point. (The two arrows in (d) need not intersect.)

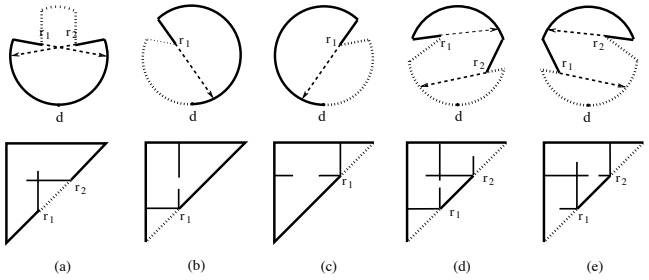


Figure 7: The reflex vertex patterns that cause trap cells bordered by G.

all possible cases where $P(d)$ is not searchable. It is easier to think of the remaining cases in terms of the trap cells that are bordered by line G. If a point on line G is on the border of a trap cell, we say that it is *unreachable*. The corresponding point on ∂P is said to be *unreachable* from d. For example, every point between points 2 and 4 on G in the SV-diagram shown in Fig. 5(b) is unreachable. In Fig. 7 we list all the reflex vertex patterns that cause unreachable sections on G, in addition to those in Fig. 6.

Theorem 2 [8] *A room $P(d)$ is searchable by two guards if and only if the door is not on the boundary of a trap cell and there is some point on G that is reachable within the SV-diagram of the room.* ■

4 A linear searchability checking algorithm

We say that a polygon P is LR-visible if we can find two points s and t on ∂P such that $\partial_{cw}(s,t)$ and $\partial_{cw}(t,s)$ are weakly mutually visible [3]. We can find in a LR-visible polygon, in linear time, a set of pairs (A_i, B_i) such that for any $s \in A_i$ and $t \in B_i$, P is LR-visible with respect to (s,t) [3]. We can also compute the non-redundant clockwise and counter-clockwise components in linear time. We first give the following lemma.

Lemma 3 [8] *If a room $P(d)$ is searchable by two guards, then P is LR-visible with respect to d and a point on ∂P.* ■

Thus if P is not LR-visible (which can be checked in linear time [3]), then room $P(d)$ is not searchable by two guards. In addition, we assume that d is on A_0, after possible relabeling.

We first consider the patterns in Fig. 6. The patterns in Fig. 6 (a), (b) and (c) are not possible, since they imply that the corresponding B_0 must be simultaneously present in two disjoint components. The pattern in Fig. 6 (d) is exactly a *deadlock*, which can be precomputed in linear time [2].

We then consider the patterns in Fig. 7. The pattern in Fig. 7 (a) involves a deadlock. As for the pattern in Fig. 7 (b), with the shortest path tree precomputed from d, we can, starting clockwise from d, find the farthest $C_{cw}(r)$ that does not contain d in linear time by checking each clockwise component [3, 1]. $\partial P_{cw}(d,r)$ is marked as unreachable. The pattern in Fig. 7 (c) is similarly dealt with.

As for the pattern in Fig. 7 (d), we notice that the corresponding B_0 should lie inside $C_{cw}(r_1)$. The relationship between r_1 and r_2 can be described by the three conditions. (1) $d \in C_{cw}(r_2)$ and $d \notin C_{cw}(r_1)$; (2) $B(r_2) \prec_d r_1$; and (3) $B(r_1) \prec_d r_2$. We look for the farthest $C_{cw}(r_1)$ clockwise from d and the farthest $C_{cw}(r_2)$ counter-clockwise from d that satisfy the above three conditions.

We preprocess the polygon with respect to A_0 and B_0 by computing the shortest path tree from d and the shortest path tree from a point $t \in B_0$ to any other vertices in linear time [1]. Note that P is LR-visible with respect to d and t.

We first consider the clockwise components on $\partial P_{cw}(d,t)$. We immediately know that we only need to consider the non-redundant ones since if there is any redundant component that satisfies the above three conditions, we can always find a non-redundant one that is even farther away from d. We move along $\partial P_{cw}(d,t)$ from d and check the non-redundant components. We only consider those that do not contain d, which can be checked in $O(1)$ time for each of them. These components have the property that if $C_{cw}(r')$ and $C_{cw}(r'')$ are two such components and $r' \prec_d r''$, then $B(r') \prec_d B(r'')$, and for any r whose clockwise component is such a component, $B(r) \in \partial P_{cw}(t,d)$. For these components, we denote their originating reflex vertices in a set called L_{cw}. We maintain a pointer P_l for traversing them, which starts at the first vertex in L_{cw} and moves clockwise from d to t.

We next consider the clockwise components on $\partial P_{cw}(t,d)$. It is obvious that this time we need to consider all the clockwise components since each of them could be possibly the farthest. We check each clockwise component as whether t is outside in $O(1)$ time using the shortest path trees from d and t [1]. For these components, we put their originating reflex vertices in a set called R_{cw}. We also maintain a pointer P_r for moving over them, which starts at $B(P_l)$ and moves clockwise

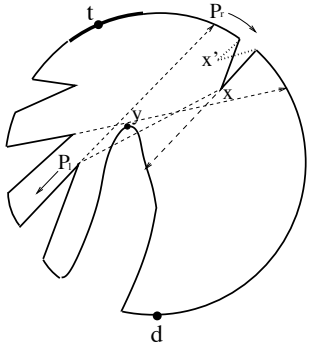


Figure 8: Illustration for the algorithm for detecting the pattern in Fig. 7 (d).

towards d.

The algorithm is described as follows, using Fig. 8 as a reference. P_r moves to the next reflex vertex x in R_{cw}. Note that if P_r goes across any backward extension point from a reflex vertex in L_{cw}, we update P_l to it.

We know that the backward extension from x can intersect neither $\partial P_{cw}(t,x)$ nor $\partial P_{cw}(x,d)$. We only need to test whether the backward extension intersects $\partial P_{cw}(d,P_l)$ or $\partial P_{cw}(P_l,t)$. If the latter is true, we give up x since $C_{cw}(x)$ contains P_l, and continue to the next element in R_{cw}. If the former is true, P_l moves to the next element in L_{cw}.

In order to find whether the backward extension from x intersects $\partial P_{cw}(d,P_l)$ or $\partial P_{cw}(P_l,t)$, we first check whether P_l and x are mutually visible. By preprocessing the polygon with respect to (d,t) in $O(n)$ time [2], this checking can be done in $O(1)$. If they are mutually visible, we only need to check whether the extension is to the right or left of the segment $\overline{xP_l}$.

Suppose that P_l and x are not mutually visible. In this case $\partial P_{cw}(B(P_l),x)$ cannot block the visibility between P_l and x. If it is so, there exists a reflex vertex on $\partial P_{cw}(B(P_l),x)$ whose clockwise component contains d and does not contain P_l. This violates the condition that x is the first candidate reflex vertex after $B(P_l)$ that P_r is exploring. Therefore, the mutual visibility between P_l and x is blocked by $\partial P_{cw}(d,P_l)$. We maintain another pointer P_x which starts from P_l and moves in the counter-clockwise direction towards d. P_x goes through each vertex and checks its visibility with x in constant time. Like before, this visibility cannot be blocked by $\partial P_{cw}(B(P_l),x)$. Therefore, P_x will eventually reach a reflex vertex y, since otherwise d is not contained in $C_{cw}(x)$. Clearly y is a reflex vertex. We associate y with P_l. We then check the backward extension from x with the segment $\overline{xy}$, from which we can easily check whether the extension intersects $\partial P_{cw}(P_l,t)$ or $\partial P_{cw}(d,P_l)$. If x satisfies the three conditions, mentioned above, we save the pair P_l and x, which indicates that $\partial P_{cw}(d,P_l)$ and $\partial P_{cw}(x,d)$ are unreachable.

We move P_l to the next reflex vertex in L_{cw} and P_r to the corresponding extension point, and the whole process repeats. Note that x is in $C_{cw}(P_l)$.

One important fact is when we repeat the above process with new P_l and P_r, we might need to move counter-clockwise P_x again. However, any previously checked polygonal chains will not be checked again. Any vertex on $\partial P_{cw}(d,t)$ will be checked at most once. In addition, P_r always advances towards d. Thus we can detect the pattern in Fig. 7 (d) in $O(n)$ time. The pattern in Fig. 7 (e) can be dealt with symmetrically.

After the unreachable parts on the boundary of P have been marked, checking whether there is any part unmarked can be done in linear time.

Theorem 4 *For a room $P(d)$ with n vertices, we can check its searchability by two guards in $O(n)$ time.* ∎

5 Conclusion

The main contribution of this paper is to present a simple characterization of searchable rooms by two guards and propose a linear algorithm to check the characterization. In designing the algorithm, we have made use of visibility diagrams and some previous results. Our algorithm can be easily adapted to tackle the room search by one searcher with a flashlight.

References

[1] B. Bhattacharya and S. K. Ghosh. Characterizing LR-visibility polygons and related problems. *Computational Geometry: Theory and Applications*, 18:19–36, 2001.

[2] B. Bhattacharya, A. Mukhopadhyay, and G. Narasimhan. Optimal algorithms for two-guard walkability of simple polygons. In *Algorithms and Data Structures, 7th International Workshop, WADS 2001, Proceedings*, pages 438–449, 2001.

[3] G. Das, P. J. Heffernan, and G. Narasimhan. LR-visibility in polygons. *Computational Geometry*, 7:37–57, 1997.

[4] T. Kameda, M. Yamashita, and I. Suzuki. On-line polygon search by a seven-state boundary 1-searcher. *IEEE Transactions on Robotics*, 2006, To appear.

[5] S. M. LaValle, B. Simov, and G. Slutzki. An algorithm for searching a polygonal region with a flashlight. *International Journal of Computational Geometry and Applications*, 12(1-2):87–113, 2002.

[6] S. M. Park, J.-H. Lee, and K.-Y. Chwa. Searching a room by two guards. *International Journal of Computational Geometry and Applications*, 12(4):339–352, 2002.

[7] I. Suzuki and M. Yamashita. Searching for a mobile intruder in a polygonal region. *SIAM Journal on Computing*, 21(5):863–888, October 1992.

[8] Z. Zhang. *The Applications of Visibility Space in Polygon Search Problems.* PhD thesis, Simon Fraser University, British Columbia, Canada, Burnaby, British Columbia, Canada, 2005.