Optimal schedules for 2-guard room search

Stephen Bahun* Anna Lubiw*

Abstract

We consider the problem of searching a polygonal room with two guards starting at a specified door point. While maintaining mutual visibility and without crossing the door, the guards must move along the boundary of the room and eventually meet again. We give an $O(n^3)$ algorithm for finding a search schedule that minimizes the total distance travelled by the guards and an $O(n^6)$ algorithm that minimizes the time required for the search by solving L_1 shortest path problems among curved obstacles in a polygon.

1 Introduction

The two-guard room search problem is a variation on the problem of searching for a mobile intruder inside a polygon, introduced in [14]. The present variation was first proposed by Park et al. [13] and involves two guards walking along the boundary of a polygonal region. Given a *room* (P, d) the guards start at point d (the *door*) on the boundary of polygon P, and they are allowed to walk along the boundary of the room, initially going in opposite directions, without ever crossing the door point. The goal is for the guards to maintain mutual visibility at all times and meet again somewhere else on the boundary of the room, at which point the whole room has been searched for a mobile intruder. We consider the problem of finding a shortest or fastest *search schedule* dictating the motion of the guards subject to these conditions.

The first paper on this problem [13] gave an $O(n \log n)$ time algorithm to determine if a room can be searched. This was improved to $O(n)$ by Bhattacharya et al. [1]. Neither paper considers the time or distance required to search the room.

In an earlier variant of the problem called "searching a corridor" the final meeting point of the guards is specified as well as the initial point, and the guards may not cross either of these points. Room search is not solvable by testing all possible final meeting points, since there are rooms that can be searched only by allowing the guards to cross the final meeting point [13]. For searching a corridor, Icking and Klein [7] find a search schedule of minimum total length in time $O(n \log n + k)$ where $k \in O(n^2)$ is the size of the output. Our optimization approach is quite different.

There is also a wealth of research on versions of the problem with different numbers of guards and different types of visibility—see [8] and [10] for the case of one guard; Lavalle et al. [6] for "visibility-based pursuit-evasion" with multiple guards; and Efrat et al. [5] for a chain of k guards.

In this paper, we present an algorithm that will find optimal search schedules, if they exist, using two different notions of optimality: the shortest distance travelled and the shortest length of time required to search the room. The shortest distance schedule is found in $O(n^3)$ time and the fastest schedule is found in $O(n^6)$ time. This algorithm makes use of a search space that is a visibility diagram describing the valid positions for the two guards to maintain mutual visibility. The concept was first discussed in [8] and later modified by Zhang [15]. To maintain relevance to the current problem, we use the latter version, which assumes that two points on a single polygon edge are mutually visible.

The rest of the paper is presented as follows. In section 2 we give the background information describing the visibility diagram, its relationship to the problem, and how to create it. In section 3, we relate paths in the visibility diagram to search schedules of the room and discuss optimality of search schedules. In section 4, we examine what types of curves can appear on the visibility diagram and we discuss how the visibility diagram can be constructed. In section 5, we discuss how to solve the shortest paths problems in the visibility diagram, giving the optimal schedules. Section 6 is a summary of the work presented here.

2 Preliminaries

For any room (P, d), the *visibility diagram* [8] encodes, for any pair of points on the boundary of P, whether or not the points are mutually visible. We will consider a reduced visibility diagram from [1] as the *V-diagram*, containing the minimum amount of relevant information for the room search problem. The diagram's boundary is a right triangle with the left and top sides equal in length to the perimeter of P. With the top left corner $(0, 0)$ representing d, the x (resp. y) coordinate represents the distance along the border of P from the door in the clockwise (resp. counterclockwise) direction. Then we can associate any point (x, y) in the diagram with

*David R. Cheriton School of Computer Science, University of Waterloo, {sbahun, alubiw}@cs.uwaterloo.ca

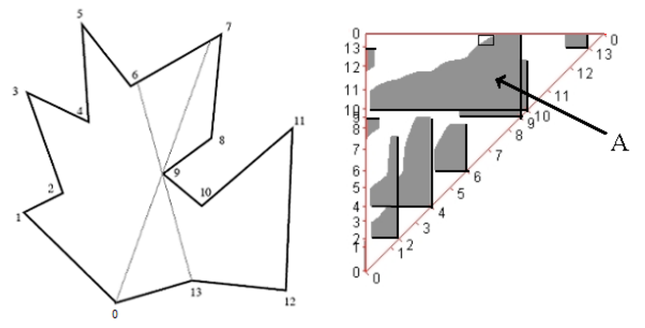

Figure 1: Room and its V-diagram from [1]

positions of the two guards on the border of P. To represent the mutual visibility of pairs of points, any point (x, y) in the V-diagram is shaded iff the corresponding positions are not mutually visible in the polygon. Figure 1 shows an example of this, where 0 is the door point. Any point on the diagonal corresponds to a meeting point of the two guards. Then any path π in the V-diagram from the top-left corner (the door) to the diagonal (the *goal*) that does not cross any shaded areas (obstacles) corresponds to a valid search schedule s of P; we say that $s = S(\pi)$.

We will assume that each guard can travel independently at varying speeds in the range [0, 1] both forwards and backwards (without crossing the door point). For any search schedule s, let $D(s)$ denote the distance travelled by the guards during s, and let $T(s)$ denote the time required for s. It should be noted that minimizing each of these may result in different search schedules. Figure 2 shows an example of a room, with a door at 0, where the two notions of optimal schedule give two different search schedules. The shortest distance is achieved if one guard travels from 0 to 2, while the other guard waits at 5, resulting in a total distance travelled equal to 24 (the perimeter of the polygon). This schedule takes over 19 time units. The quickest search schedule involves one guard travelling along 0, 5, 4, 3, and 2 while the other guard must go from 0 to b and back to a to maintain mutual visibility. This requires backtracking from b to a, which results in a total distance travelled that is greater than 24, but it takes less than 15 time units to complete the entire search.

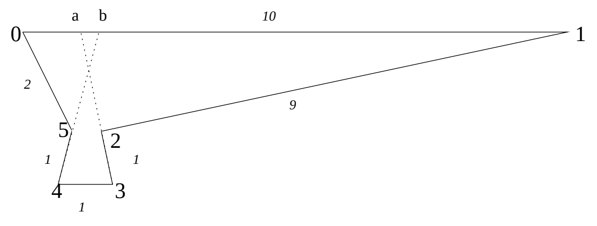

Figure 2: Optimal schedules are different

3 Correspondence Between Paths in the V-diagram and Search Schedules

The V-diagram can be used to construct optimal schedules for searching the room. We first describe a relationship between the length of a path in the V-diagram and the distance/time of the corresponding room search schedule.

Lemma 1 *Let π be a path in the V-diagram for a room and $s = S(\pi)$ be the corresponding search schedule of the room. Then $|\pi|_1 = D(s)$ and $|\pi|_\infty = T(s)$.*

Proof. This is mostly a matter of defining $D(s)$ and $T(s)$ more formally. First consider the L_1 metric. If the path π is a line segment from a to b then $|\pi|_1 = |a-b|_1 = |(a-b)_x| + |(a-b)_y|$. This represents the sum of the distances travelled by the two guards, so $|\pi|_1 = D(s)$. More generally, if π is a polygonal line joining points $a_0, \ldots, a_k$ then $|\pi|_1 = \sum_{i=1}^k |a_i - a_{i-1}|_1$. Since $D(\cdot)$ is also additive, we again have $|\pi|_1 = D(s)$. Finally, for a general curve $\pi = \pi(t), t \in [0, T]$, the length of π is defined to be $\sup(\sum_{i=1}^k |\pi(t_i) - \pi(t_{i-1})|_1$ where the sup is taken over partitions $t_0, \ldots, t_k$ of $[0, T]$. (We only consider *rectifiable curves*, where by definition the sup exists.) This is the only sensible way to define $D(s)$. Hence $|\pi|_1 = D(s)$.

Now consider the L_∞ metric. If the path π is a line segment from a to b then $|\pi|_\infty = |a-b|_\infty = \max\{|(a-b)_x|, |(a-b)_y|\}$. Since both guards have the same maximum speed of 1, assume that the guard travelling the greatest distance travels at speed 1 to minimize the time spent, which is then that guard's distance. We have $|\pi|_\infty = T(s)$. More generally, if π is a polygonal line joining points $a_0, \ldots, a_k$ then $|\pi|_\infty = \sum_{i=1}^k |a_i - a_{i-1}|_\infty$. Since $T(\cdot)$ is also additive, we again have $|\pi|_\infty = T(s)$. Finally, for a general curve π, the length is defined via a sup similar to the above and this is the only sensible way to define $T(s)$. Hence $|\pi|_\infty = T(s)$.

$\square$

Corollary 2 *The search schedule requiring the shortest amount of distance to travel along the room boundary corresponds to the shortest path in the L_1 metric from the door to the goal in the V-diagram. The quickest search schedule for a room (P, d) corresponds to the shortest path in the L_∞ metric from the door to the goal in the V-diagram.*

4 V-diagram Construction

We now discuss the creation of the V-diagram by examining the nature of the diagram.

Theorem 3 *The border of each obstacle in the V-diagram is piecewise hyperbolic.*

We begin with an intermediate result that is proved in the appendix.

Lemma 4 *When a single reflex vertex obstructs visibility between two edges in the polygon, the curve on the corresponding region of a shaded portion of the V-diagram is hyperbolic.*

We now prove Theorem 3.

Proof. Each obstacle in the V-diagram is a union of *barriers*, where a barrier represents the guard positions blocked by one reflex vertex. A barrier has a horizontal and vertical side determined by the two edges adjacent to the reflex vertex [15]. The remainder of the barrier boundary is defined by the pairs of points on the polygon that are connected by a line that also goes through the reflex vertex. For example, obstacle A in Figure 1 is composed of two barriers, associated with reflex vertices 9 and 10. The box in the figure shows the portion of the barrier where vertex 9 would obstruct the view between guards on edges (6,7) and (0,13).

From the above lemma, we know that a single reflex vertex obstructing visibility between two edges of the polygon corresponds to a hyperbolic portion of a barrier in the V-diagram. When the endpoint of one edge is reached, a new edge is reached, so a new hyperbolic portion of the barrier is begun. So each obstacle is piecewise hyperbolic. □

To construct the V-diagram we need to accurately describe all of the obstacles. To do this, we must, for each of the reflex vertices:

1. Find the two points at which the projections of the adjacent edges through the vertex first intersect the polygon again. This takes $O(n)$ time.

2. Starting at one of these points and working along the polygon towards the reflex vertex, for each edge of the polygon that is encountered, find the curve of the V-diagram representing the pairs of points on the polygon that have a line of visibility through the reflex vertex. This takes $O(n)$ time.

Therefore, with $O(n)$ reflex vertices, the exact visibility diagram can be described in $O(n^2)$ time.

5 Finding Shortest Paths in V-diagram

We have now reduced the problem of finding optimal search schedules with respect to distance and time to the problem of finding shortest paths in the L_1 and L_∞ metrics among curved obstacles in the plane that are piecewise hyperbolic.

We note the following two properties of L_1 and L_∞ shortest paths:

1. Between any two points there is a shortest L_1 path that is rectilinear. This remains true in the presence of curved obstacles, except in the situation – that doesn't arise for us – where the shortest path travels between two abutting curved objects.

2. The L_1 and L_∞ norms are related by a linear mapping; in particular we can find shortest L_∞ paths by rotating the plane and its obstacles by 45° and scaling, and then finding shortest L_1 paths. This is justified in [9]. Thus it suffices to find shortest L_1 paths, either among the original obstacles, or among the obstacles rotated by 45°.

There is considerable work on finding shortest L_1 paths among polygonal obstacles in the plane [2, 12]. There is also work on "curvilinear" computational geometry [4] which has led to shortest path algorithms among "splinegons" [11]. However, there appears to be no solution in the literature to finding shortest L_1 paths among curved obstacles. There are two basic approaches to solving this problem. The continuous Dijkstra approach is used by Mitchell [12] for polygonal obstacles. Alternatively, the problem may be modelled as a graph shortest path problem [3]. The former approach will likely lead to a more efficient solution, but we will simply claim a polynomial time algorithm via modelling the problem on a graph.

Lemma 5 *There exists a shortest path among obstacles in the L_1 metric such that the only points on the path that intersect obstacle boundaries are local extreme points of the obstacles in either the horizontal or vertical direction.*

Proof. The proof is included as an appendix. □

Now to find the shortest path, we will convert the relevant points from the V-diagram (extreme points of obstacles and the door point) into vertices of a weighted graph, where edge weights represent the L_1 distances between the points.

The graph has an edge between a pair of V-diagram points iff there is a path between the points that is monotone in both x and y, and the pair of points is minimal in the sense that no such path goes through a third V-diagram point. The weight of the edge is the L_1 length of the path. Let N be the number of extreme points. Since there are $O(n)$ barriers each with $O(n)$ extreme points, thus N is $O(n^2)$. In the case where the obstacles are not rotated by 45° (the original L_1 case) we obtain a tighter bound of $N \in O(n)$ because the curved parts of the barriers in the V-diagram are monotone in both x and y, so there are only 3 extreme points per barrier.

To represent the goal line of the V-diagram, we extend the V-diagram and add one more point g, equidistant from every point on the goal line; this point is the reflection of the door point in the goal line. Then any path from the door point to g goes through the goal line followed by a direct path to g. Thus the minimal path to the goal line is contained within a minimal path to g.

Now we need to determine which edges to include in the graph. For every pair of points ρ and ψ, add an edge between them in the graph iff no other extreme points lie in the interior of the rectangle with ρ and ψ at opposite corners and no obstacles cross the entire length or width of this rectangle. This can be accomplished in $O(N^3)$ time using brute force.

On the constructed graph, Dijkstra's algorithm finds a shortest path from the door vertex to g in $O(N^2)$ time. This provides us with the points along the shortest path in the V-diagram. To find the actual path in the V-diagram, it suffices to find, for any two consecutive points ρ and ψ a path that is monotone in x and y. This can be done by following the boundary of the rectangle containing ρ and ψ, detouring around any encountered obstacles. Correctness follows from our rule about which edges were added to the graph.

Finally, the V-diagram path from the door to the goal yields the room search schedule.

6 Conclusion

We have shown how to use a visibility diagram of a room to create an optimal schedule to search the room with two guards. By solving a shortest L_1 path problem among curved obstacles, a shortest distance schedule can be found in $O(n^3)$ time and a fastest route schedule can be found in $O(n^6)$ time. While this technique has not yielded remarkably fast running times, it is a novel approach that may be applicable to finding optimal schedules for other mobile guard problems.

References

[1] Binay Bhattacharya, John Z. Zhang, Qiaosheng Shi, and Tsunehiko Kameda. An optimal solution to room search problem. In *Proceedings of the 18th Canadian Conference on Computational Geometry (CCCG'06)*, pages 55–58, 2006.

[2] Danny Z. Chen, Kevin S. Klenk, and Hung-Yi T. Tu. Shortest path queries among weighted obstacles in the rectilinear plane. *SIAM Journal of Computing*, 29(4):1223–1246, 2000.

[3] K. Clarkson, S. Kapoor, and P. Vaidya. Rectilinear shortest paths through polygonal obstacles in $O(n(\log n)^2)$ time. In *SCG '87: Proceedings of the third annual symposium on Computational geometry*, pages 251–257, New York, NY, USA, 1987. ACM Press.

[4] David P. Dobkin and Diane L. Souvaine. Computational geometry in a curved world. *Algorithmica*, 5(3):421–457, 1990.

[5] Alon Efrat, Leonidas J. Guibas, Sariel Har-Peled, David C. Lin, Joseph S. B. Mitchell, and T. M. Murali. Sweeping simple polygons with a chain of guards. In *11th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 927–936, 2000.

[6] Leonidas J. Guibas, Jean-Claude Latombe, Steven M. LaValle, David Lin, and Rajeev Motwani. A visibility-based pursuit-evasion problem. *International Journal of Computational Geometry and Applications*, 9(4-5):471–494, 1999.

[7] Christian Icking and Rolf Klein. The two guards problem. *International Journal of Computational Geometry and Applications*, 2(3):257–285, 1992.

[8] Steven M. LaValle, Borislav H. Simov, and Giora Slutzki. An algorithm for searching a polygonal region with a flashlight. In *SCG '00: Proceedings of the sixteenth annual symposium on Computational geometry*, pages 260–269, 2000.

[9] D. T. Lee and C. K. Wong. Voronoi diagrams in L_1 (L_∞) metrics with 2-dimensional storage applications. *SIAM Journal on Computing*, 9(1):200–211, 1980.

[10] Jae-Ha Lee, Sang-Min Park, and Kyung-Yong Chwa. Searching a polygonal room with one door by a 1-searcher. *International Journal of Computational Geometry and Applications*, 10(2):201–220, 2000.

[11] E. A. Melissaratos and Diane L. Souvaine. Shortest paths help solve geometric optimization problems on planar regions. *SIAM Journal of Computing*, 21(4):601–638, 1992.

[12] J. S. B. Mitchell. L_1 shortest paths among polygonal obstacles in the plane. *Algorithmica*, 8(1):55–88, 1992.

[13] Sang-Min Park, Jae-Ha Lee, and Kyung-Yong Chwa. Searching a room by two guards. *International Journal of Computational Geometry and Applications*, 12(4):339–352, 2002.

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

[15] Zhong Zhang. *Applications of Visibility Space in Polygon Search Problems.* PhD thesis, Simon Fraser University, August 2005.

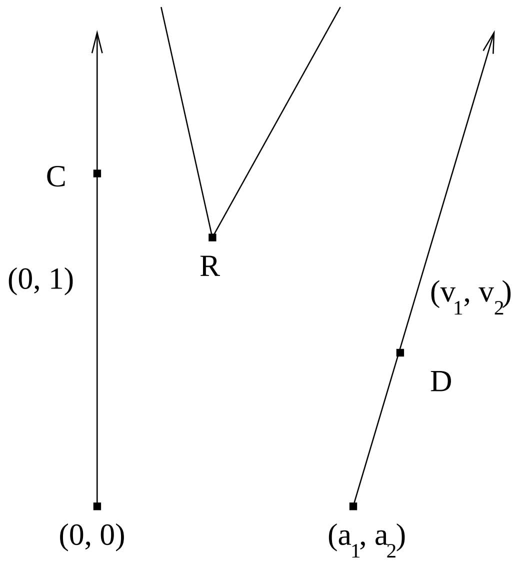

Figure 3: Reflex vertex

A Proof of Lemma 4

Without loss of generality, assume that one of the endpoints of one of the edges is $(0,0)$, and the direction along that side is represented by the vector $(0,1)$. Then let $R = (r_1, r_2)$ be the reflex vertex, let (a_1, a_2) be one endpoint of the other edge, and let (v_1, v_2) be the normalized vector representing the direction along the other edge. Let $C = (0,0) + s(0,1)$ be any point along the first edge and let $D = (a_1, a_2) + t(v_1, v_2)$ be any point along the second edge. The barrier of the shaded region has a curve corresponding to the points along the edges where the visibility line along the edges goes through P (that is, when CR and DR are parallel). Then by treating the points as points in three dimensions, we can use the vector cross product to decide that two points along the edges give a point on this curve whenever

$CR \times DR = 0$
$\iff (((0,0) + s(0,1)) - (r_1, r_2)) \times ((a_1, a_2) + t(v_1, v_2) - (r_1, r_2)) = 0$
$\iff (-r_1, s - r_2) \times (a_1 + tv_1 - r_1, a_2 + tv_2 - r_2) = 0$
$\iff -r_1 a_2 - r_1 t v_2 + s a_1 + s t v_1 - s r_1 + r_2 a1 + r_2 t v_1 = 0$
$\iff -s(a_2 + tv_2 - r_2) = -r_1 a_2 - r_1 t v_2 + r_2 a_1 + r_2 t v_1$
$\iff s = \frac{r_1 a_2 + r_1 t v_2 - r_2 a_1 - r_2 t v_1}{a_1 + t v_1 - r_1}$

Thus the relationship between the movement along one edge and the movement of the projection of the line of sight along another edge of the polygon is given by the equation of a hyperbolic curve.

B Proof of Lemma 5

By note (1) at the beginning of section 5, there is a shortest path that is rectilinear. Suppose π is a shortest rectilinear path that contains obstacle intersection

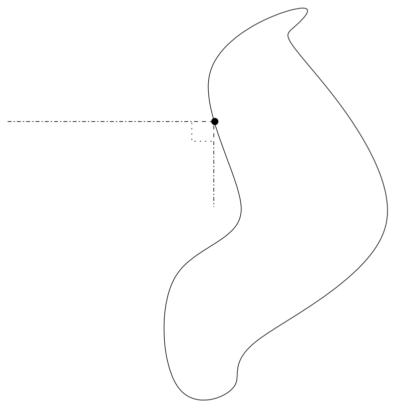

Figure 4: Intersection of path and obstacle

points that are not local extreme points. Let $q = \pi(t)$ be the first such point on π.

If π does not change direction at q then q is a local extreme point. Otherwise π makes a right angle turn at q, and we can alter the path as shown in 4. The revised path has the same length, and if the detour is small enough, the revised path intersects no obstacles. Applying this inductively yields the desired path.