Capturing an Evader in a Polygonal Environment with Obstacles

Deepak Bhadauria and **Volkan Isler**
Department of Computer Science and Engineering
University of Minnesota
{bhadau,isler}@cs.umn.edu

Abstract

We study a pursuit-evasion game in which one or more cops try to capture a robber by moving onto the robber's current location. All players have equal maximum velocities. They can observe each other at all times. We show that three cops can capture the robber in any polygonal environment (which can contain any finite number of holes).

1 Introduction

Pursuit-evasion games in which one or more pursuers try to capture an adversarial evader have numerous robotics applications such as surveillance and search-and-rescue. Due to its practical importance, there has been significant interest in solving pursuit-evasion games in complex environments. Such environments are usually represented either topologically with graphs or geometrically using polygons.

One of the earliest games on graphs is the game introduced by Parsons [Parsons, 1978]. In this game, pursuers try to capture an arbitrarily fast evader on a graph by "clearing" the vertices and surrounding the evader. In the cops-and-robbers game [Nowakowski and Winkler, 1983], the evader and the pursuers move in turns. At each turn, players can move to an adjacent vertex. The pursuers try to capture the evader by moving onto its current vertex.

The visibility-based pursuit-evasion game [Suzuki and Yamashita, 1992; Guibas *et al.*, 1999] takes place in a polygon. Similar to Parson's game, the evader in this game is arbitrarily fast. The pursuers try to capture the evader by finding it (i.e., by establishing line of sight). The lion-and-man game [Alonso *et al.*, 1992; Sgall, 2001] can be considered as a geometric version of the cops-and-robbers game. In the original version of this game, a lion tries to capture a man in a circular arena by moving on to his current location. Both players have equal maximum velocities. In this paper, we study this game in polygonal environments with obstacles, and show that three pursuers can capture the evader in any polygonal environment.

In addition to its theoretical importance, our result provides a pursuit strategy in a number of practical settings. One example is the scenario studied by [Vieira *et al.*, 2009]. In this scenario, pursuers try to capture an evader in an environment where a sensor network is deployed. The sensor network provides the location of the evader to the pursuers. It also facilitates communication among the pursuers. Since the underlying domain in this game is polygonal, our results imply that only three pursuers would suffice regardless of the floor plan. Another example in which the evader's location is revealed to the pursuers is when an aerial vehicle supports the ground-based pursuit team as illustrated in Figure 1.

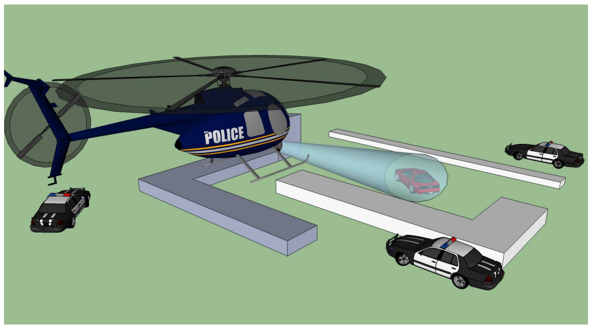

Figure 1: A practical application of our result: if the location of the evader is provided to the cops (by the helicopter), three cops can capture the robber in any polygonal environment. We ignore the non-holonomic constraints.

We build on the results by Aigner and Fromme who study the cops-and-robbers game and show that three cops can capture the robber on any planar graph [Aigner and Fromme, 1984]. It turns out that this result does not directly translate into pursuit-evasion in a polygonal environment due to the continuous nature of the underlying domain. One might think that this difficulty can be overcome simply by discretizing the domain perhaps by placing a grid with resolution comparable to the step-size of the players. Unfortunately, playing the game on such a grid does not guarantee capture in the original problem since the evader can not be restricted to stay on the vertices of the grid. Further, computing strategies using this representation can be extremely costly in terms of running time. To see this, imagine scaling the environment while keeping the step size constant. This way, the size of the grid can be made arbitrarily large. Instead, an algorithm whose complexity is proportional to the geometric parameters of the environment (e.g. number of vertices of the polygon) is desirable. One might try to circumvent this issue by playing the game on a graph that captures the topology of the underlying polygon (e.g. the triangulation graph or its dual). However, since the evader can not be restricted to stay on the edges of this graph as well, it can not be captured using the strategy by Aigner and Fromme.

Since there is no obvious way of using the graph-based pursuit strategy for pursuit in polygonal domains, we developed a new strategy which captures the continuous nature of the domain. In our strategy, the pursuers restrict the evader to smaller and smaller polygons with fewer number of obstacles or vertices. Once the evader is restricted to a simply-connected polygon (with no holes) it gets captured by using a modification of the strategy presented by Isler et al. [Isler *et al.*, 2005]. We show that capture takes a finite number of steps.

The paper is organized as follows. In Section 2 we describe the game model. We present preliminary results adapted from previous work in Section 3. Section 4 gives the details of our three-cop strategy and its analysis. We conclude the paper in Section 5 with directions for future research.

2 Game Model

The game is played in a simple polygon P which has finite number of polygonal obstacles (or holes) in it. An instance of the game is shown in Figure 2. There are two types of players in the game: cops and a robber. The following rules are used to play the game: At the beginning of the game, each cop picks a location in P. Next, the robber chooses its location in P. Afterwards, the game is played in alternate turns. All cops move simultaneously in a turn. If anytime during the game a cop can move to the current position of the robber, then the cops win the game.

We assume that the maximum speeds of all the cops and the robber are the same. We normalize the distances such that a cop or a robber can move at most one unit in its turn. The players know each others' positions throughout the game. The cops can communicate with each other at all times during the game. By communicating they can coordinate their moves.

3 Preliminaries

In this section, we present two results that will be used in our strategy. The first result is from [Aigner and Fromme, 1984] and shows that a single cop can guard a shortest path in the sense that whenever the robber crosses it, it will be captured. The second result is adapted from [Isler *et al.*, 2005] and shows that a single cop can capture the robber in any simply-connected polygon (a simply-connected polygon is a polygon with no holes).

[Aigner and Fromme, 1984] introduced the concept of "guarding" a path in graphs. Their result applies to any metric space, which gives us the following lemma.

Lemma 3.1 (Guarding a shortest path [Aigner and Fromme, 1984]). *In a finite number of moves, a single cop C can position itself on any shortest path π in such a way that for any point $x \in \pi$, the distance between x and C is less than or equal to the distance between x and the robber R's location. Afterwards, C can move in such a way that if R crosses π, it will be captured by C.*

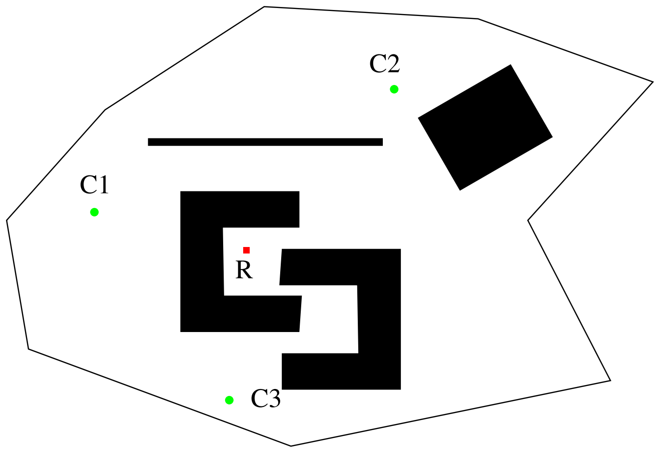

Figure 2: A polygon P with obstacles. Cops $C1$, $C2$ and $C3$ try to capture robber R.

Note that this guarding strategy involves an *initialization phase* in which the cop positions itself on π to make sure that no point on π is closer to the robber than the cop. During the initialization phase, the robber *can* cross π. However, once the initialization phase is over, the robber can no longer cross π without being captured. The length of the initialization phase is bounded by the length of π.

[Isler *et al.*, 2005] studied the visibility-based version of the cops-and-robbers game in simply-connected polygons. In their model, a cop can see the robber only if the line segment connecting the two players does not intersect the boundary of the polygon. They showed that a single cop can locate the robber, and two cops can capture the robber in any simply-connected polygon. In the two-cop strategy, one cop starts from an arbitrary point o and moves so that it stays on the shortest path between the robber's current location and o. Further, whenever the cop moves, its distance from o increases at least by a constant amount. Since the cop can not see the robber when it is occluded from his field of view, the second cop is used to determine the motion direction when the robber is not visible. They also bound the number of searches necessary. Since in our model the players know each other's locations at all times, the second cop is not necessary, giving us the following result:

Lemma 3.2 (Capture in a simply connected polygon [Isler *et al.*, 2005]). *A single cop can capture the robber in any simply-connected polygon in finite time.*

Using these two results as subroutines, we now present a strategy for three cops to capture the robber in any polygonal environment (possibly with obstacles).

4 Three-Cop Strategy

In this section we present a strategy for three cops to catch a robber in finite time. We will divide the cops' strategy into rounds. In each round, the cops will coordinate their moves and restrict the robber to a smaller polygon.

The game starts with the cops and the robber situated at their initial positions. Then the robber makes the first move and the cops respond with their moves. In each round cops pick a shortest path in the polygon to guard [1]. Guarding this path essentially partitions the current polygon, and restricts the robber to a region which is smaller than the robber's region at the beginning of the round.

In any round, the cops' strategy will ensure that at most two paths will need to be guarded by the cops. Therefore in any round, at most two cops will be guarding these two paths (by Lemma 3.1) and one cop will be free. In each subsequent round, the free cop will guard a new path and relieve one of the other cops.

Before presenting the full strategy, we describe two types of moves. In each round cops will perform either a *slicing move* and/or an *obstacle move*. Each of the two moves is a sequence of steps taken by a single cop. Before presenting the details, we introduce the notation we will use throughout the paper.

At round i, the robber will be restricted to a polygonal region P_i. We denote the boundary of P_i by δP_i. Let $n(P_i)$ be the total number vertices in P_i (including the obstacle vertices). The boundary δP_i will consist of at most two shortest paths, π_1 and π_2, each guarded by a dedicated cop. The rest of the boundary will either consist of a portion of δP, the original polygon's boundary, or the boundaries of the obstacles. Hence if robber tries to escape from P_i it has to cross either π_1 or π_2 which will result in capture by Lemma 3.1. We label the vertices of π_1 and π_2 in the order they are encountered while traversing δP_i in clockwise direction. Without loss of generality, let $\pi_1 = u_1, \ldots, u_k$ and let $\pi_2 = u_l, \ldots, u_m$.

At the end of each round, the strategy will maintain the following invariants:

1. $n(P_i) > n(P_{i+1})$, the number of vertices in P_{i+1} are strictly smaller than the number of vertices in P_i.
2. $P_{i+1} \subset P_i$, i.e., the new polygon is a subset of the previous one.
3. the paths guarded by the cops forming the boundary of P_{i+1} are both the shortest paths in P_{i+1}.

We are now ready to present the two types of moves and analyze their properties.

4.1 Obstacle Move

This move is performed when an obstacle is touching either π_1 or π_2. First consider the case where there is an obstacle touching exactly one of π_1 or π_2. Suppose there is an obstacle touching π_1 but not π_2 as shown in Figure 3-Top. In this case, the obstacle move is performed by finding a shortest path from u_1 to u_k in the interior of P_i excluding the points on π_1. Let this new path be π_3. The third cop starts guarding π_3. Since the robber can be either between π_3 and π_1 or between π_3 and π_2, one of the cops from π_1 or π_2 will be free and the robber will be restricted to a smaller region.

[1]The move will be over when the initialization phase of Lemma 3.1 is complete.

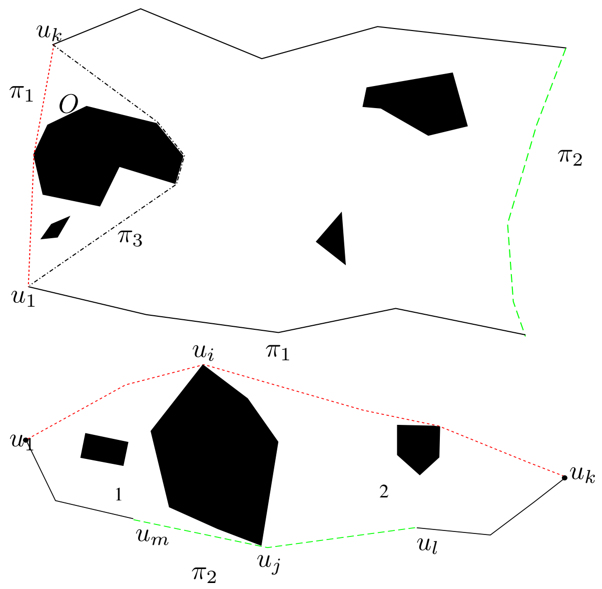

Figure 3: Two possible obstacle moves. **Top:** π_3 is the new guarded path. **Bottom:** An obstacle move where new paths to guard are portions of the old paths.

In the remaining case, there is an obstacle which is touching the boundary of P_i in multiple connected regions (see Figure 3-Bottom). This means that the interior of P_i is composed of multiple connected components. In this case the robber is already restricted to the connected component it lies in. The obstacle move is to simply switch to guarding the portion of π_1 and π_2 which are part of the boundary of this region. For example, on the right side of the Figure 3, if the evader is in region 2 then the new π_1 (resp. π_2) is the path from u_i to u_k (resp. u_l to u_j).

Lemma 4.1. *After an obstacle move, all the invariants mentioned above are maintained.*

Proof. We verify that each invariant is maintained.

1. In each obstacle move, we remove an obstacle from P_i and at least one vertex of this obstacle is not included in P_{i+1}.
2. An obstacle move divides P_i into at least two regions, and we pick one. Therefore, $P_{i+1} \subset P_i$.
3. π_3 is a shortest path in P_{i+1}. So are π_1 and π_2. Hence, the two guarded paths in P_{i+1} are both shortest paths.

$\square$

4.2 Slicing Move

The slicing move is used to restrict the robber to a smaller polygon when no obstacle touches the guarded paths. In a slicing move two points u_a and u_b are picked from δP_i such that u_a (respectively u_b) lies on the boundary portion between u_k and u_l (respectively u_1 and u_m). We compute a shortest path between u_a and u_b and use the third cop to guard this path as shown in Figure 4. Note that if there is no path between u_a and u_b in P_i, this means that u_a and u_b are in two

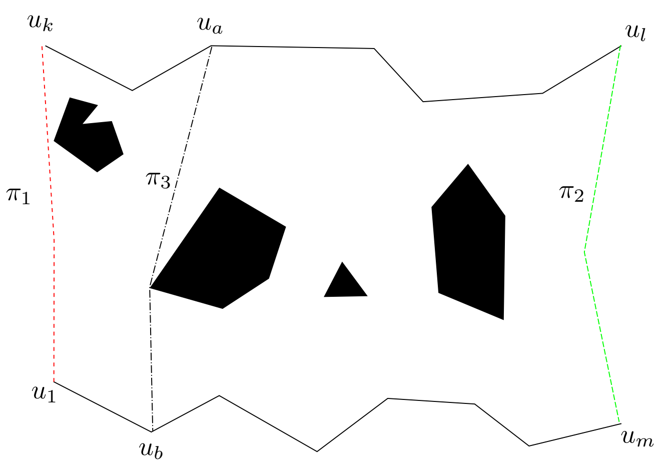

Figure 4: A slicing move. π_3 replaces either π_1 or π_2 as a path to guard depending on the location of the robber at the end of the initiliazation phase of guarding π_3.

different components (i.e. P_i is disconnected). This can happen only when there is an obstacle whose boundary is touching δP_i in multiple connected regions making it disconnected. In this case we can use the obstacle move presented in the previous section (Figure 3-right).

We now describe how u_a and u_b are chosen. There are a few variants of the slicing move based on the number of vertices between the endpoints of π_1 and π_2 (Figures 5 and 6).

Case 1: If π_1 and π_2 share no common endpoints, π_3 is chosen as the shortest path connecting u_k and u_m (i.e. we pick u_k as u_a and u_m as u_b). This case is illustrated in Figure 5 (left).

Case 2: In the second case, π_1 and π_2 share a common endpoint (say u_k), and there is at least one vertex on the boundary between the other endpoints (u_m and u_1). In this case π_3 is chosen as the shortest path connecting $u_k = u_l$ and an arbitrary vertex between the other two endpoints. This case is illustrated in Figure 5 (right).

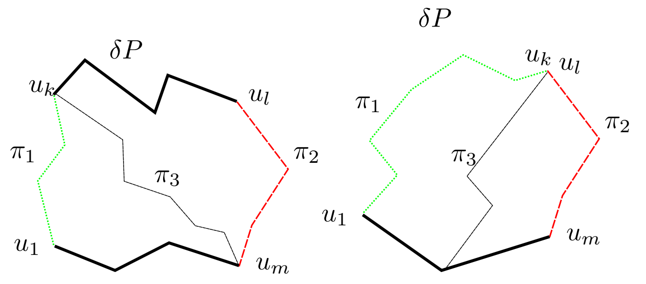

Figure 5: The first two instances of the slicing move. **Left:** The endpoints of π_1 and π_2 are different (Case 1). **Right:** The paths share one endpoint. The other end points are not adjacent (Case 2).

Case 3: In the third case, π_1 and π_2 have exactly one common endpoint and the other endpoints are adjacent (See Figure 6). Since an obstacle move is not possible, π_1 and π_2 are not touching any obstacles. In this case, π_1 and π_2 along

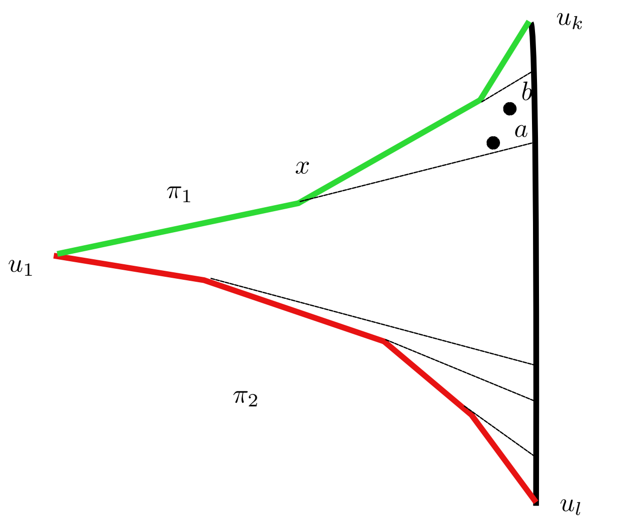

Figure 6: Case 3: π_1 and π_2 share one endpoint and the other endpoints are adjacent.

with the polygonal edge, (u_k, u_l) form a structure called a *funnel* [Guibas *et al.*, 1987]. The common end-point (u_1 in Figure 6) is the apex of the funnel. Both π_1 and π_2 are inwardly convex: when walking from the apex to u_k, one would always turn locally left. This is because π_1 is a shortest path and no obstacle is touching it from the inside. Therefore, if there was a right turn (i.e. an inward convex vertex), one can find a shorter path from u_1 to u_k which is a contradiction. A symmetric argument holds for π_2.

When there are no obstacles inside the funnel, the inward convex structure of π_1 and π_2 yields a simple partition of the funnel which can be used for computing shortest paths easily. The partition is obtained by extending each edge of π_1 and π_2 toward the edge (u_k, u_l) as shown in Figure 6. Suppose edge e on π_1 was extended to form the boundary of a partition cell. The shortest path from u_1 to point a in this partition cell continues along π_1 until it leaves e, followed by a line segment from the last vertex of e to a. For example, in Figure 6, the shortest paths from u_1 to a and b continue along π_1 until x followed by the line segment xa and xb respectively. We refer the last vertex on the boundary as the *corner* vertex of a point. Hence x is the corner vertex of both a and b.

We now describe the slicing move for this case. If there are no obstacles inside the funnel, the third cop can capture the robber by Lemma 3.2. Otherwise, we show that there exists a point on the edge (u_l, u_k) whose shortest path from u_1 touches an obstacle.

Remove all the obstacles from the funnel and compute the partition described above. We start from the leftmost partition and move toward right. For each partition we order all the obstacle vertices in that partition in anti-clockwise direction with respect to their corner vertex. We extend the line segment from the corner vertex to the first obstacle vertex in this ordering until it hits edge (u_l, u_k). In Figure 7, for partition tx_2x_3 we extend the line segment from t to the first vertex in the ordering until it hits (u_l, u_k) at o. Therefore the shortest path π_3 from u_1 to o touches the obstacle. The third cop guards this path. We now consider the part of the funnel the evader is restricted to. If the part contains no obstacles, the free cop guarding either π_1 or π_2 can capture the robber (Lemma 3.2). Otherwise, π_3 is touching an obstacle. Therefore, we can perform an obstacle move.

Observe that in forming π_3 we introduced a new vertex in P_i (at o in Figure 7). However, in computing P_{i+1} we removed at least two vertices (either u_k or u_l plus the obstacle vertex touching π_3). Hence the invariant $n(P_{i+1}) < n(P_i)$ is maintained.

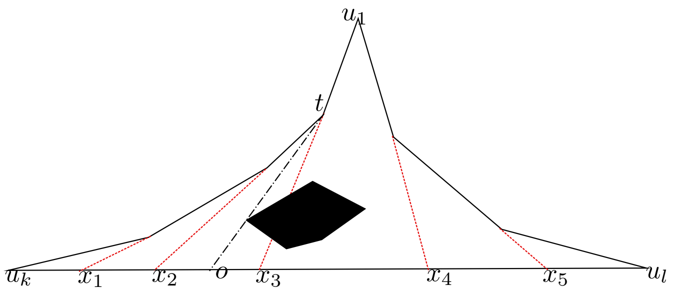

Figure 7: A funnel with its partition. There exists a point o on the boundary such that the shortest path from u_k to o touches the boundary.

Case 4: The only remaining case is when π_1 and π_2 have common endpoints at both ends. Since π_1 and π_2 are both shortest paths, it must be that $\pi_1 = \pi_2$ and the evader has already been captured, otherwise we get a contradiction with the fact that neither π_1 nor π_2 is touching any obstacles.

Among the four cases mentioned above, case 4 yields capture. In remaining cases 1-3, u_a and u_b can be chosen accordingly and a free cop can be assigned to guard the shortest path π_3 between u_a and u_b (recall that such a path exists because an obstacle move is not possible). The path π_3 partitions P_i to at least two regions. The robber can be either between π_3 and π_1 or between π_3 and π_2, therefore one of the cops from π_1 or π_2 can be relieved from guarding the path.

We now prove a technical lemma.

Lemma 4.2. *The paths π_1 and π_2 can be chosen so that they do not share an edge.*

Proof. Let us assume that π_1 and π_2 overlap which implies that they share at least one edge. Also, there must be at least one edge in π_1 which is not in π_2, otherwise guarding π_2 is redundant. We also know that both π_1 and π_2 lie on the boundary of P_i. But since P_i is simple, the shared edges will include one of the two edges at the end of the two paths. Therefore, without loss of generality, we can assume that the last edge of π_1 and the first edge of π_2 is shared. Equivalently, let $\pi_1 = u_1, \ldots, u_l, u_k$ and $\pi_2 = u_l, u_k, \ldots, u_m$, i.e., π_1 and π_2 share the edge (u_l, u_k).

Since π_1 is a shortest path, a path $\pi_1' = u_1, \ldots, u_l$ will also be a shortest path between u_1 and u_l. If the two cops pick paths π_1' and π_2 to guard, the edges guarded will be the same as before, but the paths will be non-overlapping. Hence, as a result, we can always find π_1 and π_2 such that they don't share a common edge. $\square$

By Lemma 4.2, π_1 and π_2 will not overlap. Further, cases 1-4 above cover all possible non-overlapping cases. Therefore we have:

Corollary 4.3. *If an obstacle move does not exist, either the robber is captured or a slicing move exists.*

For case 3, we have already shown that $n(P_{i+1}) < n(P_i)$. In all other cases, similar to the proof of Lemma 4.1, it can be verified that the slicing move maintains the invariants giving us the following lemma.

Lemma 4.4. *After a slicing move, all the invariants are maintained.*

4.3 Complete Strategy and Analysis

We are now ready to describe the full strategy. At the beginning of the game, two cops pick two separate edges and guard them as π_1 and π_2. Afterwards, the cops perform an obstacle move if an obstacle is touching π_1 or π_2, or an obstacle is touching the boundary of P_i in two connected regions. If an obstacle move is not possible, a slicing move is performed which is always possible by Corollary 4.3.

We now present our main result which shows that the sequence of moves described above result in capture in finite number of steps.

Theorem 4.5. *Three cops can capture the robber in a finite number of steps in any polygon with a finite number of obstacles.*

Proof. Suppose the step size of the cops and the robber is one. Let P be the initial polygon and n be the number of vertices of P. After each round, the robber is restricted to a subset of the area before the move. Further, the new area has at least one less vertex than the previous area. Therefore, the number of rounds is bounded by the number of vertices (including obstacle vertices).

Next we bound the length of each round. Each slicing or obstacle move is comprised of picking a shortest path π in P_i and guarding it. Hence, the length of each round is bounded by the time to reach π followed by the initialization phase of guarding it. Each of these quantities is bounded by the diameter of P_i (i.e. the longest shortest path in P_i). Therefore, the length of round i bounded by $2diam(P_i)$. $\square$

Is there an upper bound on $diam(P_i)$ in terms of the parameters of P? While it is not easy to obtain a tight bound on this quantity, it is easy to see that it is bounded by the area A of P because shortest paths are non-intersecting. This suggests an upper bound of $2nA$ for the capture time.

For a given polygon, this upper bound is likely to be loose when the shortest paths are much shorter than the area A. However, the capture time can be as large as A in general: imagine a long and thin rectangular environment with cops starting on one end and the robber on the other. We also note that the diameter of P_i can be larger than the diameter of P. We leave the problem of obtaining a tight bound on capture time as an interesting problem for future research.

5 Conclusion

We studied a new variant of the cops and robbers game, and showed that three cops can capture the robber in any polygonal environment. In our model the players have equal maximum velocities and have access to each others' locations at all times. As noted earlier, an immediate question for future research is to obtain a tight bound on capture time.

An important extension for future work is to relax the assumption about the information available to the players. For example, if the cops can obtain information about the robber's location only when they establish line-of-sight visibility, how many cops are necessary? How does the outcome change if there are communication limitations?

Another challenging extension is to study the case when the cops are subject to non-holonomic motion constraints as in the case of a car chase.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants 0916209, 0917676 and 0936710. We thank Shaun Gosse, Joseph Pipp and members of the Robotic Sensor Networks Lab for useful discussions, and their help with the figures.

References

[Aigner and Fromme, 1984] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984.

[Alonso *et al.*, 1992] L. Alonso, A.S. Goldstein, and E.M. Reingold. " Lion and Man": Upper and Lower Bounds. *INFORMS Journal on Computing*, 4(4):447, 1992.

[Guibas *et al.*, 1987] L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R.E. Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2(1):209–233, 1987.

[Guibas *et al.*, 1999] L.J. Guibas, J.C. Latombe, S.M. LaValle, D. Lin, and R. Motwani. A visibility-based pursuit-evasion problem. *International Journal of Computational Geometry and Applications*, 9(4/5):471, 1999.

[Isler *et al.*, 2005] V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion in a polygonal environment. *IEEE Transactions on Robotics*, 21(5):875–884, 2005.

[Nowakowski and Winkler, 1983] R. Nowakowski and P. Winkler. Vertex to vertex pursuit in a graph. *Discrete Math.*, 43(2):235–239, 1983.

[Parsons, 1978] T. Parsons. Pursuit-evasion in a graph. *Theory and applications of graphs*, pages 426–441, 1978.

[Sgall, 2001] J. Sgall. Solution of David Gale's lion and man problem* 1. *Theoretical Computer Science*, 259(1-2):663–670, 2001.

[Suzuki and Yamashita, 1992] I. Suzuki and M. Yamashita. Searching for a mobile intruder in a polygonal region. *SIAM Journal on computing*, 21:863, 1992.

[Vieira *et al.*, 2009] M.A.M. Vieira, R. Govindan, and G.S. Sukhatme. Scalable and practical pursuit-evasion with networked robots. *Intelligent Service Robotics*, 2(4):247–263, 2009.