Capturing an Evader in Polygonal Environments with Obstacles: The Full Visibility Case*

Deepak Bhadauria[1], Kyle Klein[2], Volkan Isler[1], and Subhash Suri[2]

[1] Department of Computer Science, University of Minnesota, MN 55455, USA, {bhadau, isler}@cs.umn.edu

[2] Department of Computer Science, UC Santa Barbara, CA 93106, USA, {kyleklein, suri}@cs.ucsb.edu

Abstract

Suppose an unpredictable evader is free to move around in a polygonal environment of arbitrary complexity that is under full camera surveillance. How many pursuers, each with the same maximum speed as the evader, are necessary and sufficient to guarantee a successful capture of the evader? The pursuers always know the evader's current position through the camera network, but need to physically reach the evader to capture it. We allow the evader the knowledge of the current positions of all the pursuers as well—this accords with the standard worst-case analysis model, but also models a practical situation where the evader has "hacked" into the surveillance system. Our main result is to prove that *three* pursuers are always *sufficient and sometimes necessary* to *capture* the evader. The bound is independent of the number of vertices or holes in the polygonal environment.

1 Introduction

Pursuit-evasion games provide an elegant setting to study algorithmic and strategic questions of exploration or monitoring by autonomous agents. Their mathematical history can be traced back to at least 1930s when Rado posed the now-classical Lion-and-Man problem [16]: a lion and a man in a closed arena have equal maximum speeds; what tactics should the lion employ to be sure of his meal? The problem was settled by Besicovitch who showed

*Research of Klein and Suri on this paper was supported in part by the National Science Foundation Grant IIS-0904501. Research of Bhadauria and Isler on this paper was supported in part by the National Science Foundation Grants CCF-0916209 and IIS-0917676. This paper combines results from two independently discovered algorithms, whose preliminary versions appeared at the 25th Conference on Artificial Intelligence [14] and the 20th International Joint Conference on Artificial Intelligence [2].

that the man can escape regardless of the lion's strategy [16]. An important aspect of this pursuit-evasion problem, and its solution, is the assumption of continuous time: each player's motion is a continuous function of time, which allows the lion to get arbitrarily close to the man but never capture him. If, however, the players move in discrete time steps, taking alternating turns but still in continuous space, the lion can always catch the man.

A rich literature on pursuit-evasion problems has emerged since these initial investigations, and the problems tend to fall in two broad categories: discrete space, where the pursuit occurs on a graph, and continuous space, where the pursuit occurs in a geometric space. Our focus in this paper is on the latter: visibility-based pursuit in a polygonal environment in two dimensions. There exist simply-connected n-gons that may require $\Omega(\log n)$ deterministic pursuers in the worst-case to detect a single, arbitrarily fast moving evader, and $O(\log n)$ pursuers also always suffice for all n vertex simple polygons [8]. When the polygon has h holes, the number of necessary and sufficient pursuers turns out to be $O(\sqrt{h} + \log n)$ [8]. However, these results hold only for detection of the evader, *not for the capture.*

For capturing the evader, it is reasonable to assume that the pursuers and the evader all have the same maximum speed. Under this assumption, it is shown by Isler et al. [11] that two pursuers with line-of-sight vision can capture the evader in a *simply-connected* polygon using a *randomized* strategy whose expected search time is polynomial in n and the diameter of the polygon. When the polygon has holes, no non-trivial upper bound is known for capturing the evader. For instance, we do not even know if $O(h)$ pursuers are able to capture the evader. Because visibility-based pursuit allows *unbounded* line-of-sight visibility regardless of the distance, it is unclear how to map a detection strategy to a capture strategy[1].

There are two fundamental issues inherent in pursuit evasion: *localization*, which is purely an informational problem, and *capture*, which is a problem of planning physical moves. In this paper, we study the question: how complex is the capture problem *if the evader localization is available for free*? In other words, suppose the pursuers have complete information about the evader's current position, how much does it help them to capture the evader?

Besides being a theoretically interesting question, the problem is also a reasonable model for many practical settings. Given the rapidly dropping cost of electronic surveillance and camera networks, it is now both technologically and economically feasible to have such monitoring capabilities. These technologies enable cheap and ubiquitous detection and localization, but in case of intrusion, a physical capture of the evader is still necessary. For instance, the scenario studied in [22] requires pursuers to capture an evader in an environment instrumented with a sensor network. The sensor network provides the location of the evader to the pursuers and facilitates communication among the pursuers. Our results immediately imply that three pursuers suffice regardless of the shape of the floor plan in their application.

[1]Indeed, one can modify the argument of Isler et al. [11] to show that if the pursuer can see the evader at all times, then a single pursuer is able to capture the evader in a *simply-connected* polygon using a deterministic strategy; we give more details about this in the appendix.

1.1 Our Contributions

Our main result is that under the full visibility setting, *three pursuers* are always sufficient to capture an equally fast evader in a polygonal environment with holes, using a *deterministic* strategy. Complementing this upper bound, we also show that there exist polygonal environments that require at least three pursuers to capture the evader even with full information.

We present two different algorithmic strategies for our main result, one called *Minimal Path Strategy* and the other *Shortest Path Strategy*. These were discovered independently by two teams, Bhadauria-Isler [2] and Klein-Suri [14] around the same time. This paper combines the main results of those two papers. The former (Minimal Path Strategy) uses the visibility graph of the original polygon, and deploys pursuers along the first, second and third shortest paths in this graph to trap the evader in progressively smaller sub-polygons (Section 3). The latter (Shortest Path Strategy) operates in the continuous domain, and guards a carefully chosen shortest path so as to trap the evader in a smaller polygonal region (Section 5). Despite their high-level similarity, the two algorithms differ significantly in details, and offer independent insights into the problem, motivating us to include them both in this joint paper.

The bound on capture time, which is asymptotically the same for both strategies, is independent of the number of the holes of the polygon, although the capture time depends on both n and the diameter of the polygon.

1.2 Related Work

There is an enormous literature on pursuit evasion and related problems [1, 3, 5, 9, 12, 13, 15, 18, 19, 20, 21]. A recent survey on search and pursuit-evasion research in robotics can be found in [4].

The research tends to fall into two distinct categories: geometry-based and graph-based. The former assumes a continuous model of space, typically a polygon, while the latter assumes a discrete graph model where agents move along edges. The graphs provide a very general setting but can suffer from two shortcomings: one, the generality leads to weak upper bounds and, two, they fail to model many restrictions imposed by the geometry of physical world. Thus, for instance, in general graphs the worst case number of cops required to capture a robber is known to be as large as $\Omega(\sqrt{n})$, and computing the minimum number of cops needed is EXP-TIME complete [6].

In visibility-based pursuit, a seminal paper [8] shows that $\Theta(\log n)$ pursuers are both necessary and sufficient in worst-case for a *simply-connected* n-vertex polygon. Most of the existing work in polygon searching, however, is on *detection* and not capture. The only relevant result on capture is by Isler et al. [11] showing that in *polygons without holes* a single pursuer can achieve detection and two pursuers with line-of-sight visibility can achieve capture. When the environment has holes, it is not even known how many pursuers are sufficient to capture an evader, even though a tight bound of $\Theta(\sqrt{h} + \log n)$ for detection is known. In one important aspect, polygon searching is fundamentally different from graph searching: *re-contamination* is unavoidable in polygons [8], in general, while graphs can always be searched *optimally* without re-contamination [15].

Our work bears some resemblance to, and is inspired by, the result of Aigner and Fromme [1] on planar graphs, showing that graph searching on planar graph requires 3 cops. In that work, the graph is unweighted, does not deal with Euclidean distances, and require players to move to only neighboring nodes. Unlike the graph model, our search occurs in continuous Euclidean plane, and players can move to any position within distance one. Thus, while our bounds are similar, the proof techniques and technical details are quite different.

2 The Problem Formulation

We assume that an evader e is free to move in a two-dimensional closed polygon P, which has n vertices and h holes. A set of pursuers, denoted $p_1, p_2, \ldots$, wish to capture the evader. All the players have the same maximum speed, which we assume is normalized to 1. The bounds in our algorithm depend on the number of vertices n and the diameter of the polygon, $\mathrm{diam}(P)$, which is the maximum distance between any two vertices of P under the shortest path metric.[2]

For the sake of notational brevity, we also use e to denote the current position of the evader, and p_i to denote the position of the ith pursuer. We model the pursuit-evasion as a continuous space, discrete time game: the players can move anywhere inside the polygon P, but they take turns in making their moves, with the evader moving first. In each move, a player can move to any position whose shortest path distance from its current position is at most one; that is, within *geodesic disk* of radius one. On the pursuers' move, all the pursuers can move simultaneously and independently. We say that e is successfully captured when some pursuer p_i becomes collocated with e.

In order to focus on the complexity of the capture, we assume a complete information (full visibility) setup: each pursuer knows the location of the evader at all times. We also endow the evader the same information, so e also knows the locations of all the pursuers. In general, neither side knows the future moves of the opponents, although our result holds *even if the evader knows all the future moves of the pursuers.* In addition, both sides know the environment P. We begin with a high level description of the minimal path strategy, followed by its technical details and proof of correctness in the next section.

3 The Minimal Path Strategy

We show that three pursuers, denoted p_1, p_2, p_3, can always capture an evader using a deterministic strategy, regardless of the evader's strategy and the geometry of the environment. The minimal path strategy is to progressively trap the evader in an ever-shrinking region

[2] We assume that the *area* quantity of the polygon is at least as large as the *diameter* of the polygon, which can be always ensured through an appropriate scaling, if needed. We give a more precise argument later in the paper. This assumption helps us frame the bounds using the diameter alone.

of the polygon P. The pursuit begins by first choosing a path Π_1 that divides the polygon into sub-polygons (see Figure 1(a))—we will use the notation P_e to denote the sub-polygon containing the evader. We show that, after an initialization period, the pursuer p_1 can successfully guard the path Π_1, meaning that e cannot move across it without being captured.

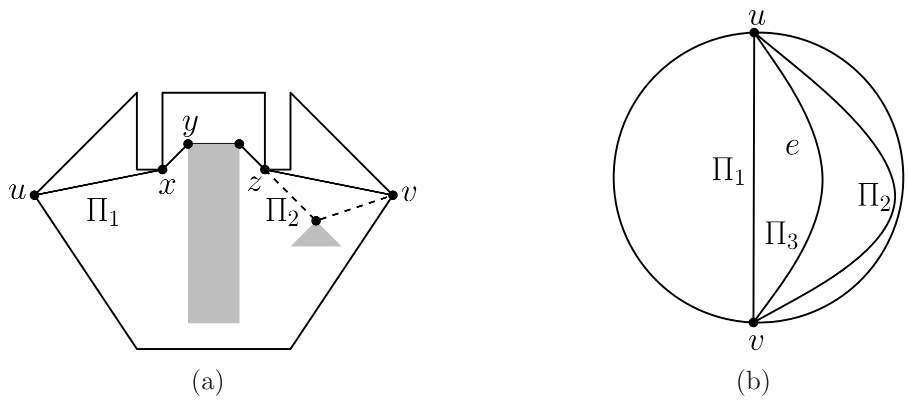


Figure 1: (a) A polygonal environment with two holes (a rectangle and a triangle). xy is a visibility edge of $G(P)$, while xz is not. Π_1 and Π_2 are the first and the second shortest paths between anchors u and v. The figure (b) illustrates the main strategy of trapping the evader through three paths.

Figure 1(b) illustrates the overall strategy: in a general step, the sub-polygon P_e containing the evader is bounded by two paths Π_1 and Π_2, satisfying a geometric property called *minimality*, each being guarded by a pursuer. We then choose a third path Π_3 splitting the region P_e into two non-empty subsets. If both regions have holes, then we argue that the pursuer p_3 can guard Π_3, thereby trapping e either between Π_1 and Π_3 (Figure 1(b)), or between Π_2 and Π_3, in which case the pursuit iterates in a smaller region. If Π_3 is not guardable within one of the regions, then we show that the pursuer p_3 can *evict* the evader from this region, forcing it into a smaller region (as measured by the number of vertices) where the search resumes.

3.1 Visibility Graphs and Path Guarding

In order for this strategy to work, the paths Π_i need to be carefully chosen and must satisfy certain geometric conditions, which we briefly explain. First, although the pursuit occurs in continuous space, our paths will be computed from a *discrete* space, namely, the *visibility graph* of the polygon. The visibility graph $G(P)$ of a polygon P is defined as follows: the nodes are the vertices of the polygonal environment (including the holes), and two nodes are joined by an edge if the line segment joining them lies entirely in the (closed) interior of the polygon. (In other words, the two vertices joined by an edge must have line of sight visibility.) This *undirected* graph has n vertices and at most $O(n^2)$ edges. We assign each edge a *weight* equal to the Euclidean distance between its two endpoints. See Figure 1(a) for an example.

One can easily see that, given two vertices u and v of P, the *shortest path* from u to v in $G(P)$ is also the shortest Euclidean path constrained to lie inside P. (The shortest Euclidean path has corners only at vertices of $G(P)$.) However, we cannot make such a claim for the *second*, or in general the kth, shortest path—one can create an infinitesimal "bend" in the shortest path Π_1 to create another path that is arbitrarily close to the first shortest path but does not belong to $G(P)$. Therefore, we will only consider paths that belong to $G(P)$ and are "combinatorially distinct" from Π_1—that is, they differ in at least one visibility edge. However, even then the kth shortest path between two nodes can exhibit counter-intuitive behavior. For instance, while in graphs with non-negative weights the first shortest path is always loop-free, the *second*, or more generally kth, shortest path can have loops—this may happen if repeatedly looping around a small-weight cycle (to make the path distinct from others) is cheaper than taking a different but expensive edge [10]. Therefore, we will consider only shortest loop-free paths. One of our technical lemmas proves that these paths are also *geometrically* non-self-intersecting. (This is obvious for the shortest path Π_1 but not for subsequent paths.) In addition, we argue that these paths also satisfy a key geometric property, called *minimality*, which allows a pursuer to guard them against an evader.

4 Proof of Sufficiency of 3 Pursuers

We begin with the discussion of how a single pursuer can guard a path in P, trapping the evader on one side. We then discuss the technically more challenging case of guarding the second and the third paths. In order to guarantee that a path in P can be guarded, it must satisfy certain geometric properties. We begin by introducing two key ideas: a *minimal path* and the *projection* of an evader on a path. In the following, we use the notation $d(x, y)$ to denote the shortest path distance between points x and y. When we require that distance to be measured within a subset, such as restricted to a path Π, we write $d_\Pi(x, y)$. That is, $d_\Pi(x, y)$ is the length of path Π between its points x and y. Occasionally, we also use the notation $\Pi(x, y)$ to denote subpath of Π between points x, y. We use the notation $x \prec y$ to emphasize that the point x precedes y on the path Π: that is, if Π is the path from node u to node v, then $x \prec y$ means that $d_\Pi(u, x) < d_\Pi(u, y)$. The following property is important for patrolling of paths.

Definition 1. *(***Minimal Path:***) Suppose Π is a path in P dividing it into two sub-polygons, and P_e is the sub-polygon containing the evader e. We say that Π is* minimal *with respect to P_e if, for all points $x, z \in \Pi$ and $y \in (P_e \setminus \Pi)$, the following holds:*

$$d_\Pi(x, z) \ \leq \ d(x, y) + d(y, z)$$

Intuitively, a minimal path cannot be shortcut: that is, for any two points on the path, it is never shorter to take a detour through an interior point of P_e. (This is a weak form of triangle inequality, which excludes detours only through points contained in P_e.) The next definition introduces the projection of the evader on to a path, which is an important concept in our algorithm.

Definition 2. *(* **Projection:** *) Suppose* Π *is a path in* P *dividing it into two sub-polygons, and* P_e *is the sub-polygon containing the evader* e*. Then, the* projection *of* e *on* Π*, denoted* e_π*, is a point on* Π *such that, for all* $x \in \Pi$*,* e *is no closer to* x *than is* e_π*.*

Thus, if a pursuer is able to position itself at the projection of e at all times, then it guarantees that the evader cannot cross the path without being captured. With these definitions in place, we now discuss how to guard the first path Π_1.

4.1 Guarding the First Path

We choose two vertices u and v on the outer boundary of P, and call them *anchors*. We let Π_1 be the shortest path from u to v in $G(P)$; this is also the shortest Euclidean path between u and v constrained to lie inside the environment. Our first observation is that this path Π_1 is always minimal.

Lemma 1. *The path* Π_1 *between* u *and* v *is minimal.*

Proof. For the sake of contradiction, suppose there are two points $x, z \in \Pi_1$ that violate the minimality. Let the point $y \notin \Pi_1$ be the witness of this violation, namely, $d(x, y) + d(y, z) < d_{\Pi_1}(x, z)$. But then Π_1 can be shortened with the subpath $\Pi_1(x, z)$, contradicting the fact that Π_1 is the shortest u, v path. □

The following lemma shows that the projection of e is always exists for a minimal path.

Lemma 2. *Suppose* Π *is a minimal path between the anchor nodes* u *and* v*. Then, for every position of the evader* e *in* P_e*, a projection* e_π *exists.*

Proof. Let us first consider the more interesting case where $d_\Pi(u, v) \geq d(u, e)$. In this case, we claim that the point z at distance $d(e, u)$ along Π is a projection of e. Indeed, for any point $x \in \Pi$ such that $z \prec x$, the condition $d_\Pi(z, x) > d(e, x)$ leads to a violation of the minimality of Π, as follows:

$$d_\Pi(u, x) \;=\; d_\Pi(u, z) + d_\Pi(z, x) \;=\; d(u, e) + d_\Pi(z, x) \;>\; d(u, e) + d(e, x)$$

Similarly, for any point x where $x \prec z$, the condition $d(x, e) < d_\Pi(x, z)$ also leads to a violation:

$$d(u, e) \;\leq\; d_\Pi(u, x) + d(x, e) \;<\; d_\Pi(u, x) + d_\Pi(x, z) \;=\; d_\Pi(u, z)$$

which is a contradiction because $d(u, e) = d_\Pi(u, z)$.

On the other hand, if $d_\Pi(u, v) < d(u, e)$, then we choose v as the projection. In this case, the argument is identical to the second case above: $\forall x \prec v$, $d(x, e) \geq d_\Pi(x, v)$, and thus v is a projection.

□

The next lemma shows how a pursuer can guard a minimal path. Whenever we refer to the projection, we mean the unique point chosen by Lemma 2, that is, the point on Π at $d(u, e)$ from u, or v, whichever is closer.

Lemma 3. *Suppose* Π *is a minimal path between the anchors* u, v *in* P*, and a pursuer* p *is located at the current projection of* e*. Suppose on its turn the evader moves from* e *to* e'*. Then, the pursuer* p *can either capture the evader or relocate to the new projection* e'_π *in one move.*

Proof. First, suppose that the new position e' is on different side of the path Π than e, namely, the evader crosses the path, say, at a point z. Because the evader can move at most distance one, we have the inequality $d(e, z) + d(z, e') \leq 1$. On the other hand, since p is located at the projection of e before the move, $d_\Pi(p, z) \leq d(e, z)$. Therefore, the new position of the evader e' is within distance one of p, and the pursuer can capture the evader on its move.

If the evader does not cross Π, and moves to a position e' on the same side of the path, let e'_π be the projection of e', as defined in Lemma 2. Because the evader moves distance at most one further from u or at most one closer to u, it must satisfy $d(e_\pi, e'_\pi) \leq 1$, and so p can relocate from e_π to e'_π in one move. □

Before proceeding further, we make a minor technical digression, to establish that any path guarded by pursuers can be bounded by the *area* of the polygon. The strategy of progressively trapping the evader within smaller sub-polygons brings out a somewhat counterintuitive property of polygon divisions: *a sub-polygon can have a larger diameter than the original polygon.* Figure 2 shows an example where the diameter of the shaded sub-polygon P' is larger than the original environment. This complicates the time complexity analysis of our pursuit strategy because it depends on the length of paths that are guarded. We resolve this dilemma by arguing these path lengths cannot exceed the area of the original environment, which in turn is bounded by $\text{diam}(P)^2$. Of course, diameter is a one-dimensional quantity, while area is a two-dimensional quantity, but we only care about their numerical magnitudes. We show the required inequality by choosing an appropriate *scale* (units) for the environment, as shown in the following lemma.

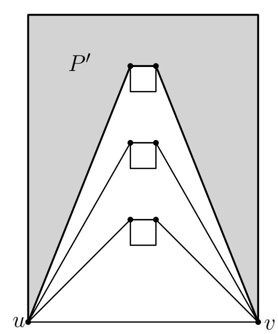


Figure 2: Example depicting a shaded sub-polygons P' with diameter larger than $\text{diam}(P)$.

Lemma 4. *Suppose* Π *is a* u, v *path in sub-polygon* P' *of* P*. Then, by applying a suitable rescaling of units we can always guarantee* $d_\Pi(u, v) \leq \text{diam}(P)^2$*.*

Proof. If $d_\Pi(u,v) \leq area(P')$, then the lemma holds trivially, because $area(P') < area(P) \leq \text{diam}(P)^2$. Therefore, assume that $d_\Pi(u,v) > area(P')$. By a simple *rescaling of the units*, we can get the desired reverse inequality, as follows. Suppose we rescale the unit of measurement from 1 to $1+\alpha$. This increases the area of a triangle by a factor of $(1+\alpha)^2$, while a segment only increases in length by a factor of $1+\alpha$. Therefore, a suitably large choice of α will always ensure that the polygon's area exceeds the length of Π, because the former grows by a factor of $(1+\alpha)^2$ while the latter grows linearly. In particular, if $(1+\alpha)^2 \cdot area(P') \geq (1+\alpha) \cdot d_\Pi(u,v)$, we obtain $\alpha \leq \frac{d_\Pi(u,v)}{area(P')} - 1$, and therefore any choice of $\alpha > \frac{d_\Pi(u,v)}{area(P')} - 1$ will suffice. □

With this technical lemma, we can assume throughout the rest of the paper that $d_\Pi(u,v) \leq diam(P)^2$ always holds. The following lemma shows that within $O(\text{diam}(P)^2)$ a pursuer p can either reach the current projection of e or capture it.

Lemma 5. *Suppose Π is a minimal path between anchors u, v in P, and a pursuer p is located at u. Then in $O(\text{diam}(P)^2)$ moves, p can move to e's projection.*

Proof. By Lemma 3, the projection of e can only shift by distance at most one along the path Π. Thus, p's strategy is simply to move along the path from one end to the other until it coincides with the current projection of e, or captures it. Meanwhile, if the projection ever "crosses over" the current position of p, the pursuer immediately can move to the new projection because at that moment p must be within distance one of the target location. Since p moves a distance of 1 in each turn, and Lemma 4 guarantees we can scale P such that all paths encountered have length at most $\text{diam}(P)^2$, the entire initialization phase takes at most $O(diam(P)^2)$ moves. □

4.2 Geometric Structure of Pursuer Paths

We now come to the main part of our pursuit strategy. The key idea is to progressively trap the evader in a region bounded by two minimal paths, which are guarded by two pursuers, and to use the third pursuer to further divide the current region. When the third pursuer subdivides the current region containing e, two possibilities emerge: either the third path is minimal with respect to both regions and thus guardable by the third pursuer, limiting the evader to a smaller region than before; or it is only minimal with respect to one of the regions and the other is hole-free, in which case the third pursuer uses the capture strategy for a simply-connected polygon to evict the pursuer from this region (or capture it). In order to formalize our strategy, we first show a key geometric property of the second and third shortest paths between the anchors in the visibility graph, namely, that they are non-self-intersecting, and therefore lead to well-defined closed regions.

Lemma 6. *Let Π_1 be the shortest path between two anchor points u and v on P's boundary, and focus on the sub-polygon P_e that lies on one side of Π_1. Let Π_2 and Π_3, respectively, be the second and the third simple (loop-free) shortest paths in the visibility graph $G(P_e)$ between u and v. Then, Π_2 and Π_3 are non-self-crossing.*

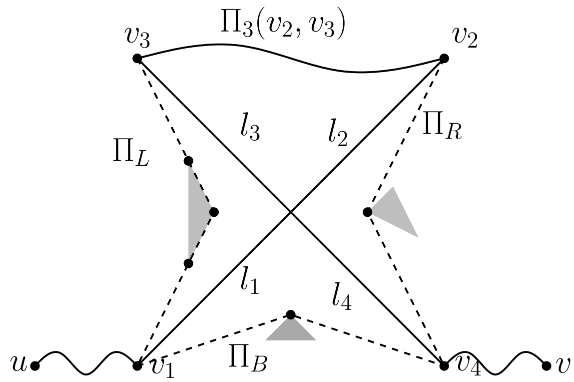


Figure 3: Non-self-crossing of shortest paths Π_1, Π_2, Π_3.

Proof. Without loss of generality, suppose the path Π_3 violates the lemma, and that two of its edges (v_1, v_2) and (v_3, v_4) intersect. See Figure 3. We first note that the intersection point cannot be a vertex of the visibility graph because otherwise the path has a cycle, and we assumed that Π_3 is loop-free. As shown in the figure, we break the segment (v_1, v_2) into l_1 and l_2, and (v_3, v_4) into l_3 and l_4. By the triangle inequality of the Euclidean metric, it is easy to see that the shortest v_1, v_3 path homotopic to the segments l_1 and l_3, denote it Π_L, will have length strictly less than $l_1 + l_3$. Similarly, define Π_R and Π_B, as paths between v_2, v_4 and v_1, v_4, respectively. Now consider the following three paths between v_1 and v_4, each contained in $G(P_e)$: $\Pi_L \cdot \Pi_3(v_3, v_2) \cdot \Pi_R$, Π_B, and the shorter of $\Pi_L \cdot (v_3, v_4)$ and $(v_1, v_2) \cdot \Pi_R$. They are all shorter than Π_3, each has one less intersection than Π_3, and at least one of them must be distinct from both Π_1 and Π_2, thus contradicting the choice of Π_3. If further intersections exist, the argument can be applied again, until all such intersections are removed. □

4.3 Shrinking, Guarding and Evicting

In a general step of the algorithm, assume that the evader lies in a region P_e of the polygon bounded by two minimal paths Π_1 and Π_2 between two anchor vertices u and v. (Strictly speaking, the region P_e is initially bounded by Π_1, which is minimal, and portion of P's boundary, which is not technically a minimal path. However, the evader cannot cross the polygon boundary, and so we treat this as a special case of the minimal path to avoid duplicating our proof argument.) We also assume that Π_1 and Π_2 only share vertices u and v; if they share a common prefix or suffix subpath, we can delete those and advance the anchor nodes to the last common prefix vertex and the first common suffix vertex. This ensures that the region P_e is non-degenerate.

The key idea of our proof is to show that, in the visibility graph $G(P_e)$, if we compute a *shortest path* from u to v that is distinct from both Π_1 and Π_2, then it divides P_e into *only* two regions, and that the evader is trapped in one of those regions. We will call this new path the *third* shortest path Π_3. Specifically, Π_3 is the simple (loop-free) shortest path from u to v

in $G(P_e)$ distinct from Π_1 and Π_2. (One can compute such a path using any of the algorithms for computing k loop-free shortest paths in a weighted undirected graph [10, 17, 23].)

Lemma 7. *The shortest path Π_3 between the anchor nodes u and v divides the current evader region P_e into two regions.*

Proof. If the path is disjoint from Π_1 and Π_2 except at endpoints, then P_e is clearly subdivided into two (possibly disconnected) regions. If Π_3 shares vertices only with Π_1 or only with Π_2, *but in multiple disjoint subpaths creating multiple regions*, then each subpath shares its first and last vertices with either Π_1 or Π_2, and thus we can replace all but one with subpaths of Π_1 or Π_2 and obtain a path no longer than Π_3. Therefore, let us suppose that Π_3 shares vertices with both the paths, and so "hops" between Π_1 and Π_2, sharing common subpaths with them, and creates three or more regions. In that case, Π_3 must leave and rejoin Π_1 and Π_2 at least once, as shown by points x, y, z in Figure 4(a). We observe that $d_{\Pi_2}(y, v)$ is no longer than $d(y, z) + d_{\Pi_1}(z, v)$, otherwise Π_2 is not the second shortest u, v path, which is a contradiction. Thus the third region can be removed by altering Π_3 to use the subpath $\Pi_2(y, v)$. (A symmetric case arises when the roles of Π_1 and Π_2 are swapped.) Thus, we conclude that Π_3 can create only two subregions. □

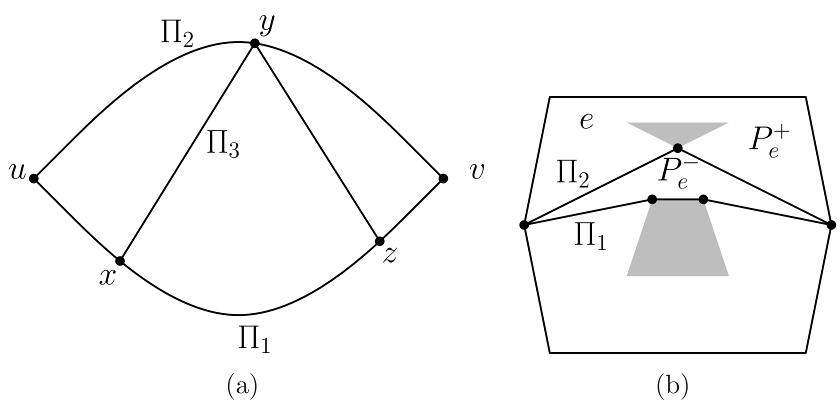


Figure 4: The left figure illustrates the proof of Lemma 7; the right figure illustrates the two subregions created by a path, Π_2 in this case.

Clearly, if P_e contains one or more holes, then at least one of the regions created by the third shortest path Π_3 also contains a hole. The following lemma argues that Π_3 is minimal with respect to such a region. (The next lemma then addresses the case when the region is hole-free.)

Lemma 8. *Suppose Π_3 divides the region P_e into two subregions P_e^+ and P_e^-, and assume that P_e^+ contains at least one hole. Then, Π_3 is a minimal path within the region P_e^+.*

Proof. Assume, for the sake of contradiction, that the minimality of Π_3 is violated for two points $x, z \in \Pi_3$. Let u' be the vertex immediately preceding the point x, possibly $x = u'$,

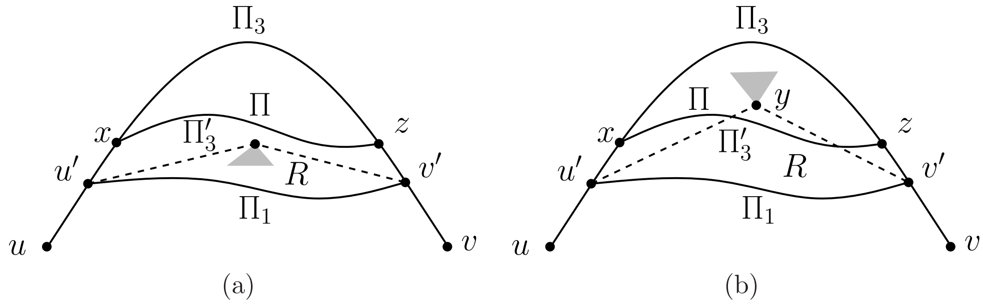


Figure 5: Illustrates the proof of Lemma 8.

and v' is the vertex immediately following z, possibly $z = v'$, on Π_3. Consider the shortest path in $G(P_e)$ from u' to v'. This path must be distinct from $\Pi_3(u', v')$, as a shortest path is necessarily minimal, while by assumption $\Pi_3(u', v')$ is not. Thus, if this path is *not* a subpath of either Π_1 or Π_2, we can immediately improve the length of Π_3 by using this subpath, thereby contradicting the choice of Π_3. Therefore, assume without loss of generality that the shortest path from u' to v' is a subpath of Π_1. Further, let Π denote the shortest path from point x to point z in P_e^+, and consider the region R bounded by $\Pi_1(u', v')$, Π and the segments (z, v') and (x, u'). If there are any holes in R then there is a distinct path Π_3' shorter than Π_3 obtained by tightening Π around those holes as shown in Figure 5(a). Thus the hole in P_e^+ must be outside R, however pick the closest vertex on a hole in P_e^+ to Π, call it y. Then a path Π_3' shorter than Π_3 can be obtained using y as shown in Figure 5(b). Thus in all cases, if P_e^+ contains a hole, Π_3 can be shortened, which contradicts its optimality. Thus Π_3's minimality cannot be violated, and the proof is complete. □

Since Π_1 and Π_2 are the two shortest paths between u and v, the region between them necessarily contains a hole: otherwise, all vertices except u and v must be reflex (within the region), which is a contradiction since every simply polygon must have at least three convex vertices. Thus, at least one of the regions created by Π_3 has a hole, and so Π_3 is minimal for that region. The region without holes must have a very special and simple structure, as shown by the following lemma, and it can be cleared using the search strategy for simply-connected polygons.

Lemma 9. *Suppose Π_3 divides the region P_e into two subregions P_e^+ and P_e^-. If Π_3 fails to be minimal with respect to P_e^+, then Π_3 has the following simple structure: two edges plus a subpath of either Π_1 or Π_2.*

Proof. Suppose Π_3 fails to be minimal in P_e^+. Then, by Lemma 8, P_e^+ is hole-free. Non-minimality means that the path can be shortcut, and so all vertices of Π_3 cannot be reflex. Let y be a vertex of Π_3 that is convex in P_e^+, and let x and z, respectively, be the predecessor and successor vertices of y. We claim that x and z are either both vertices of Π_1 or both vertices of Π_2. Suppose not. Then, the shortest path from x to z in P_e, call it Π, is shorter than $\Pi_3(x, z)$. By assumption, at least one of x and z is not in Π_1, and similarly for Π_2, thus Π cannot be a subpath of Π_1 or Π_2.

But, then the path $\Pi_3(u,x) \cup \Pi \cup \Pi_3(z,v)$ is shorter than Π_3 and distinct from Π_1 and Π_2, contradicting the choice of Π_3. Thus, x and z both belong to either Π_1 or Π_2, and assume, without loss of generality, that they belong to Π_1. Then P_e^+ is bounded by $\Pi_1(x,z)$ and the edges (x,y) and (y,z), and the proof is finished. □

Now, if both regions created by Π_3 have holes, then the minimality of Π_3 allows a third pursuer to guard this path, and the pursuit continues in one of the smaller regions. However, if one region is hole-free and Π_3 is not minimal within it, a different strategy is required. The following lemma shows how to either capture the evader in such a region, or to force the evader out of (evict) this region, while guarding Π_3 so the evader *cannot reenter* this region.

This can be accomplished by fixing an origin O in the region (say, some vertex in P), and then letting the pursuer move along the shortest path between O and the current evader position. It can be shown that the pursuer makes sufficient progress towards the evader, as described in the appendix.

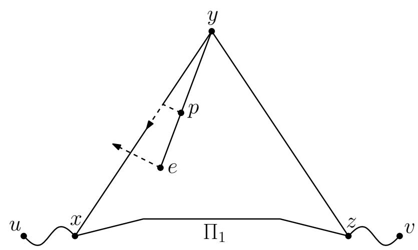


Figure 6: An illustration of the pursuer's eviction strategy. Dashed lines denote moves where e moved first.

Lemma 10. *Suppose the evader lies in hole-free region of k vertices that is bounded by Π_3 and another minimal path. If Π_3 is not minimal with respect to this region, then, in $O(k \cdot \mathrm{diam}(P)^2)$ moves, a single pursuer p can either capture the evader or force it out of the region and place itself on e's projection on the path Π_3.*

Proof. Assume, without loss of generality, that our hole-free region is bounded by a minimal path Π_1 and the path Π_3, which by Lemma 9 must consist of two edges, say, (x,y) and (y,z). The pursuer p's strategy is to move to y, and at each turn move to the point closest to e that is distance one from p and lies on the shortest y,e path, with one modification. Namely, if p's move takes it outside the region, then it moves along Π_3 toward e_π (which must exist as Π_3 is minimal with respect to the other region) until e reenters, at which point its resumes the pursuit, as depicted in Figure 6.

As the shortest path between any two vertices consists of at most two edges, this region can have diameter no larger than $2 \cdot \mathrm{diam}(P)$. Thus if e never leaves the region, then by the known result of Lemma 13 (described in the appendix), a successful capture occurs in $O(k \cdot \mathrm{diam}(P)^2)$ moves. Therefore, assume that e leaves the region at some point. Since Π_1 is minimal, the evader cannot leave the region through that path, and so assume without loss of generality that the evader crosses the segment (x, y) of Π_3. Because p always stays on the shortest path between e and y, in an unmodified pursuit p's move would cross (x, y) as well. In the modified pursuit, p stops at the point where it crosses (x, y) and advances toward the projection of e.

We note that the projection of e is within distance one of where e crossed (x, y). As a result, because p crossed (x, y) at a point closer to y than e, if e_π lies on the subpath $\Pi_3(p, v)$, then p can reach e_π in one move, and Π_3 is guarded and we are done. Otherwise, p need simply advance forward along Π_3 toward e_π. If e never re-enters the hole free region, then by Lemma 5 p will reach the projection within $O(\text{diam}(P)^2)$ moves.

In the case e re-enters the hole-free region, we note that it must do so by crossing the segment (x, p), and that for each turn e was outside the hole-free region p moved distance one along the shortest path from y to e. Thus on its next turn p can resume its pursuit, while having increased its squared distance from y by at least $1/k$, which will guarantee a successful capture occurs in $O(k \cdot \text{diam}(P)^2)$ moves should e remain within the hole-free region. Thus e may continually move back and forth between the hole-free region, but within $O(k \cdot \text{diam}(P)^2)$ moves e will either be captured, or the pursuer will successfully guard Π_3 by reaching the projection. □

We can now summarize our main result.

Theorem 1. *By following the Minimal Path Strategy, three pursuers can capture an evader in $O(n \cdot \text{diam}(P)^2)$ moves in a polygon with n vertices and any number of holes.*

Proof. Whenever a new path is introduced which is minimal with respect to both regions, the size (number of vertices) of the region P_e containing e shrinks by at least one. Thus, the number of such paths guarded during the course of the pursuit before e is captured is at most n, and the total cost of guarding them is at most $O(n \cdot \text{diam}(P)^2)$. If Π_3 is only minimal with respect to one region R, then in $O(k \cdot \text{diam}(P)^2)$ moves the evader will either be forced into R and a pursuer will guard Π_3 or the evader will be captured. In such a case, the vertices on the two bounding edges of Π_3 were not removed, thus only $k - 3$ of the k vertices were removed from P_e. When $k > 3$ the cost of removals sums to at most $O(n \cdot \text{diam}(P)^2)$. When $k = 3$, the evader is being evicted from a triangle, bounded by two edges of Π_3 which meet at a vertex y, and an edge of either Π_1 or Π_2. We bound the number of such removals by showing each vertex can only be chosen as y twice. Either y is an interior vertex of P_e, and will not be chosen again as an interior vertex (as it is now on a bounding path), or y is already on a bounding path, and y will become an anchor, and never be chosen again. Thus, there are at most $2n$ removals where $k = 3$, and their total cost is at most $O(n \cdot \text{diam}(P)^2)$.

Finally, the sub-polygon containing the evader will be reduced to a triangle. Notice this must occur, as otherwise a path Π_3 exists which would split P_e. This region clearly has diameter no larger than $\text{diam}(P)$, and thus the evader can be captured by the third pursuer in $O(\text{diam}(P)^2)$ moves with the known result of Lemma 13, for a total of $O(n \cdot \text{diam}(P)^2)$ moves over the entire pursuit. □

5 The Shortest Path Strategy

In this section we present an alternative strategy to capture the evader. In contrast to the Minimal Path Strategy which chooses the first, second and the third shortest paths in the visibility graph to trap the evader, the *Shortest Path Strategy* directly picks a shortest path in the evader's region to trap the evader in a smaller region with fewer vertices. See Figure 7.

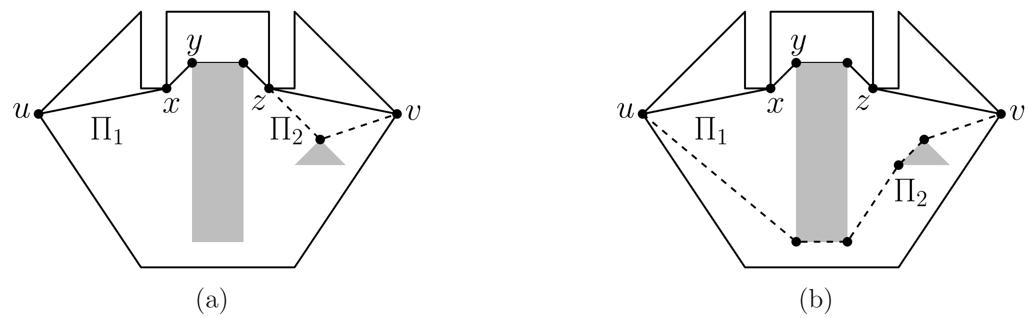


Figure 7: In (a), the next path (Π_2) chosen by the Minimal Path Strategy (Section 3). In (b), the Shortest Path Strategy using the obstacle move (Section 5.1).

A shortest path is guarded in two phases. In the initialization phase, a pursuer moves onto the evader's projection. Afterward, the pursuer stays on the projection as described in Lemma 3. Note that a shortest path in a polygon is minimal with respect to any subset of the polygon (see also Lemma 1). Hence, it can be guarded regardless of P_e.

We will divide the pursuers' strategy into rounds. In each round, the pursuers will coordinate their moves and restrict the evader to a smaller polygon by choosing two points and guarding the shortest path between them.

Before presenting the full strategy, we describe two types of moves. In each round pursuers 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 pursuer. Before presenting the details, we introduce the notation we will use for the rest of the paper.

We will use P_i to denote the the evader's region P_e at round 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 pursuer. 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 the evader tries to escape from P_i it has to cross either π_1 or π_2 which will result in capture by Lemma 3. 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 $\pi_2 = u_l, \ldots, u_m$ (See Figure 8).

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 pursuers 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.

5.1 Obstacle Move

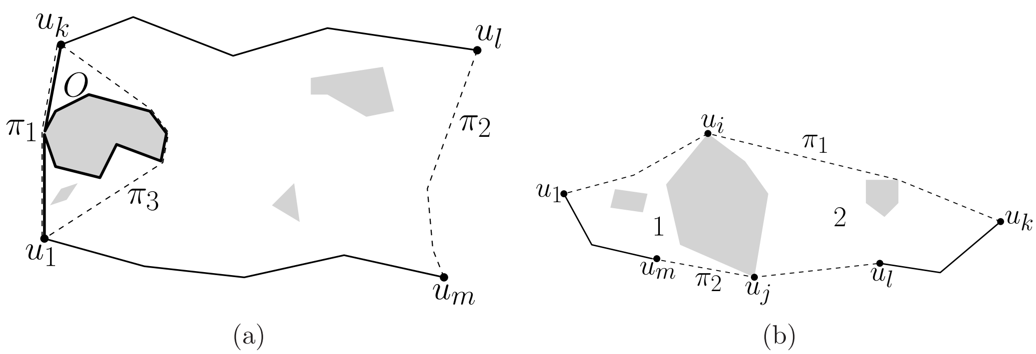


Figure 8: Two possible obstacle moves. In (a), to compute π_3, we extend the boundary ∂P_i to include ∂O (shown as the bold path). We then compute the shortest path from u_1 to u_k. In (b), an obstacle move where new paths to be guarded are portions of the old paths.

This move is performed when an obstacle O 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 8(a). 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 that touch O. To compute this path, we treat obstacles touching π_1 as part of the boundary and compute a shortest $u_1 - u_k$ path as shown in Figure 8(a). More precisely, let G be the visibility graph of P_i. We remove every edge of G which contains a point in $(\pi_1 \cap O)$. Then, we compute the shortest path from u_1 to u_k in this reduced visibility graph.

Let this shortest path be π_3. The third pursuer starts guarding π_3. Since the evader can be either between π_3 and π_1 or between π_3 and π_2, one of the pursuers from π_1 or π_2 will be free and the evader will be restricted to a smaller region.

In the remaining case, there is an obstacle which is touching the boundary of P_i in multiple points resulting in multiple connected components (see Figure 8(b)). This means that the interior of P_i is composed of multiple connected components. In this case the evader 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 8, 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 11. *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.

□

5.2 Slicing Move

The slicing move is used to restrict the evader 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 pursuer to guard this path as shown in Figure 9. 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 different components (i.e. P_i is disconnected). This can happen only when there is an obstacle whose boundary is touching δP_i at multiple locations making it disconnected. In this case we can use the obstacle move presented in the previous section (Figure 8(b)).

We now describe how u_a and u_b are chosen.

First, we observe that π_1 and π_2 can not 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 an obstacle.

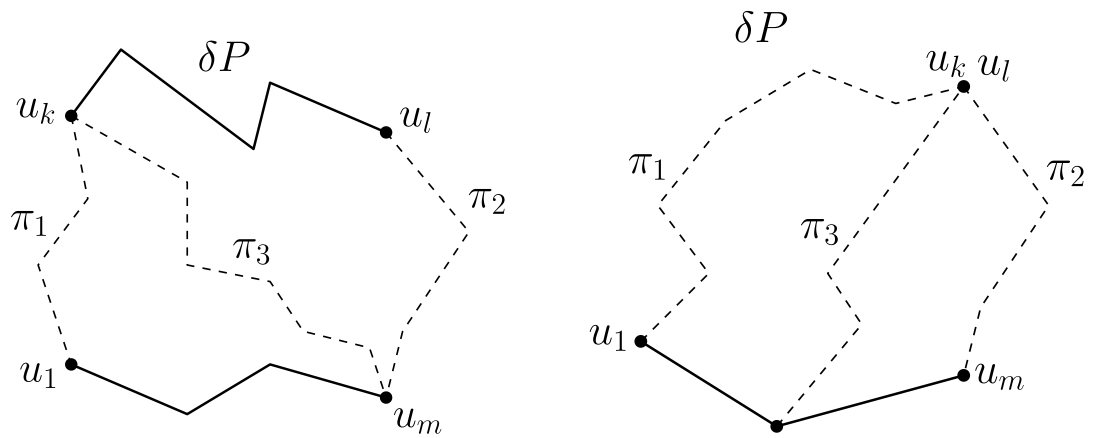
Second, if π_1 and π_2 intersect at a vertex which is not an end-point, then P_i is disconnected and the evader can be trapped in a smaller polygon simply by discarding the components which do not contain the evader.

Hence, we are left with three possibilities which yield three variants of the slicing move based on the number of boundary vertices between the endpoints of π_1 and π_2 (Figures 9 and 10).

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 9(a).

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 9(b).

Case 3: In the third case, π_1 and π_2 have exactly one common endpoint and the other endpoints are adjacent (See Figure 10). Since an obstacle move is not possible, π_1 and π_2



(a) Case 1: The endpoints of π_1 and π_2 are different

(b) Case 2: The paths share one endpoint. The other endpoints are not adjacent

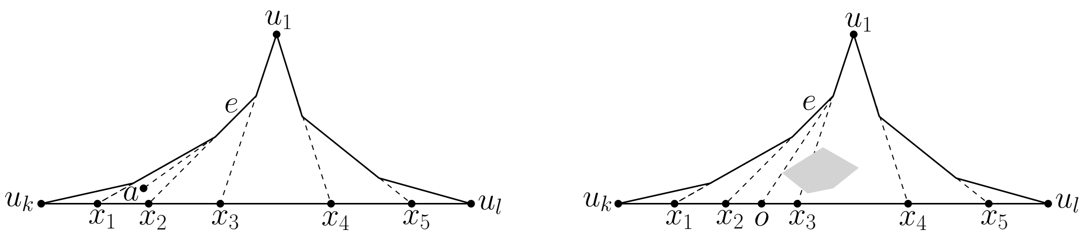
Figure 9: The first two instances of the slicing move.

are not touching any obstacles. In this case, π_1 and π_2 along with the boundary edge (u_k, u_l) form a structure called a *funnel* [7]. The common end-point (u_1 in Figure 10) 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 right. This is because π_1 is a shortest path and no obstacle is touching it from the inside. Therefore, if there was a left turn, one could find a shorter path from u_1 to u_k than π_1 which is a contradiction. A symmetric argument holds for π_2.

We now show that when the evader's current region P_i is a funnel formed by π_1, π_2 and the polygon boundary, the pursuers can trap the evader inside a triangle in such a way that at least one side of the triangle is a subset of the polygon boundary and the remaining sides are guarded by the pursuers. We start with the case when there are no obstacles inside the funnel. Even though the pursuers can readily win the game in this case by using the third pursuer and the strategy for simply connected polygons, reducing the game to a triangle yields improved capture time.

No obstacles: 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 10(a). 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. We refer the last vertex on the boundary as the *corner* vertex of a point.

The pursuers scan the funnel from left to right until they reduce it to a triangle as follows: Extend all edges on π_1 and π_2 and let $x_1, \ldots, x_m$ be the intersection of the extensions with the boundary edge (u_k, u_l) as shown in Figure 10(a). We define $x_0 = u_k$. Pursuer 3 guards the shortest path from u_1 to x_1. If the evader is to the left of π_3, we get a triangle. If the evader is to the right, we iterate by releasing the pursuer guarding the path from u_1 to u_k



(a) A funnel without obstacles is partitioned by extending the edges on π_1 and π_2.

(b) A funnel with an obstacle inside. There exists a point o on the boundary such that the shortest path from u_k to o touches the obstacle.

Figure 10: Case 3. π_1 (resp. π_2) are the shortest paths from u_1 to u_k (resp. u_l). They share one endpoint (u_1) and the other endpoints are adjacent. i.e. (u_k, u_l) is an edge on the polygon boundary.

and use him to guard the shortest path from u_1 to x_2. The pursuers continue guarding the paths from u_1 to $x_2, x_3, \ldots, x_m$ until a triangle is reached.

Note that every time the funnel is shrunk by guarding x_i, the number of vertices is reduced by one: when guarding x_i, we introduce a vertex at x_i but remove two vertices: x_{i-1} and the corner of x_i. Hence the invariant $n(P_{i+1}) < n(P_i)$ is maintained.

Obstacles inside the funnel: In this case, 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 10(b), 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 pursuer guards this path. We now consider the part of the funnel the evader is restricted to. If the part contains no obstacles, we continue as in the previous case and reduce it to a triangle. Otherwise, π_3 is touching an obstacle. We perform an obstacle move and consider this a part of the move.

Observe that in forming π_3 we introduced a new vertex in P_i (at o in Figure 10(b)). However, in computing P_{i+1} we removed at least two vertices: if the evader and the obstacle are on opposite sides of π_3, either u_k or u_l as well as all vertices on the obstacle touching π_3 are removed. If they are on the same side either u_k or u_l in addition to at least one of the obstacle vertices π_3) are removed. Hence the invariant $n(P_{i+1}) < n(P_i)$ is maintained.

We now show that a slicing move maintains all invariants.

Lemma 12. *After a slicing move, all the invariants are maintained.*

Proof. 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 11, it can be easily verified that the slicing move maintains all invariants. □

5.3 Complete Strategy and Analysis

We are now ready to describe the full strategy. At the beginning of the game, two pursuers pick two separate edges on the boundary and guard them as π_1 and π_2. Afterward, the pursuers continue with performing either an obstacle move or a slicing move until the evader region becomes a triangle as follows: If an obstacle is touching π_1 or π_2, they perform an obstacle move. If an obstacle move is not possible and P_i is not a funnel, they perform one of the slicing moves given in case 1 or case 2 until they reach a funnel. Once a funnel is reached, the pursuers reduce it to a triangle as described in case 3. When a triangle is reached, they use the third pursuer and the strategy for simply-connected polygons to capture the evader.

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

Theorem 2. *By following the Shortest Path Strategy, three pursuers can capture an evader in $O(n \cdot \mathrm{diam}(P)^2)$ moves in a polygon with n vertices and any number of holes.*

Proof. Suppose the step size of the pursuers and the evader is one. Let P be the initial polygon and n be the number of vertices of P. In order to guard a shortest path $\Pi \in P_i$, a pursuer must reach Π and move along it toward the evader's projection. Since the length of Π is bounded by $\mathrm{diam}(P)^2$ by Lemma 4, it can be guarded in $O(\mathrm{diam}(P)^2)$ steps.

At each round, at most two paths are guarded (Case 3 of a slicing move may contain an obstacle move) and at least one vertex is removed. Hence the total number of steps until the evader is trapped in a triangle is bounded $O(n \cdot \mathrm{diam}(P)^2)$. Once the evader is trapped in a triangle P_i, by Lemma 13, it can be captured in $O(3 \cdot diam(P_i)^2)$ steps. Since P_i is a triangle, the shortest paths inside P_i are the same as shortest paths inside P, hence its diameter is no greater than $diam(P)$. Therefore, the number of steps to capture the evader inside a triangle is $O(diam(P)^2)$.

To sum up, the strategy takes at most n rounds and the length of each round is $O(\mathrm{diam}(P)^2)$. Therefore the total number of steps is $O(n \cdot diam(P)^2)$. □

6 Necessity of 3 Pursuers

In this section, we complement the sufficiency of three pursuers with a lower bound. We show that any *deterministic* strategy requires at least 3 pursuers in the worst-case, and thus the upper bound of the previous section is tight.

Theorem 3. *There exists an infinite family of polygons with holes that require at least three pursuers to capture an evader even with complete information about the evader's location.*

Proof. The proof is based on a reduction from searching in *planar graphs*. In particular, consider a planar graph G, with vertices of degree 3, and no cycles of length three or four (see Figure 11(a)). Aigner and Fromme [1] proved the correctness of a simple strategy to avoid capture on such a graph, which involves moving only when a pursuer is capable of capturing it. Consider a vertex u of G with neighbors u_x, u_y and u_z. Then it is easy to

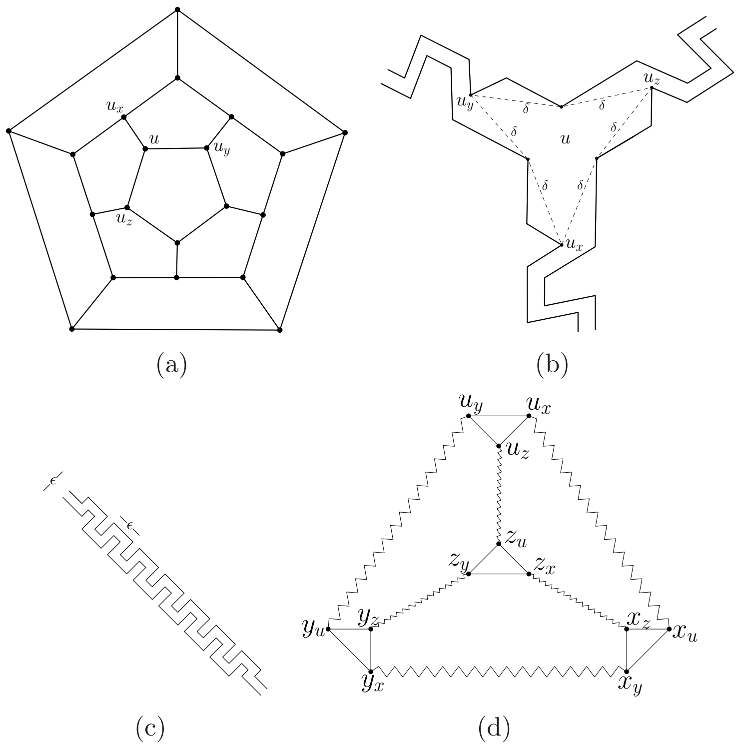


Figure 11: A planar graph with min-degree 3 and no three or four cycles (a), example constructed intersection (b), example edge construction (c), and example of corridors connecting intersections for the complete graph on four vertices (d), where jagged edges denote length $1 - 2\delta$ and straight edges 2δ.

see that no other vertex in the graph has more than one neighbor in the set $\{u_x, u_y, u_z\}$. Therefore, if there are only two pursuers, at least one of u's neighbors is *not adjacent to any pursuer*, and the evader can move to that neighbor without being captured on the pursuer's next turn. This argument repeats ad infinitum, showing that two pursuers cannot capture the evader in this graph. We now describe how to construct a polygon from G where the evader can mimic this reactive strategy and avoid capture forever against two pursuers.

Using Fary's Theorem, embed G so that each edge maps to a straight line segment. We now transform this straight-line embedding into a polygon with holes. First replace each node of G with an *intersection* shown in Figure 11(b). An intersection replacing a node u of G with neighbors x, y, z has three points labeled u_x, u_y and u_z, which we call intersection points or i-points for short. The intersection is constructed such that the shortest path between any pair of i-points (within a single intersection) has length exactly 2δ, and a shortest path through a given intersection will visit two i-points. To finish the construction, we then connect each of these intersections with corridors, such that a corridor replacing an edge from u to x will contain the i-points u_x and x_u, and by introducing artificial bends (as seen in Figure 11(c)) we can guarantee the shortest u_x, x_u path in each corridor has length

$1-2\delta$. The resulting connections between intersections for the complete graph on 4 vertices are depicted in Figure 11(d). It is easy to see that such a construction can ensure that all the corridors are non-overlapping, and by proper scaling of the environment we can meet all corridor length conditions. With this transformation, the outer face of the graph becomes the boundary of the polygon P, while each face of the plane graph becomes a hole.

We now argue that in the constructed polygon P, the evader can indefinitely avoid capture from two pursuers. To do so, the evader will move between the i-points of P, and guarantee that after each move the following invariant holds: both p_1 and p_2 are at least distance $1+2\delta$ from all i-points of e's current intersection. The game begins by each pursuer choosing a location in P, and it is easy to see that the evader can then choose some i-point such that the invariant initially holds. We then must show, that at each turn if this invariant is violated, e can move to re-establish it. By doing so we guarantee neither pursuer is ever closer than 2δ to e, and thus e can indefinitely avoid capture.

Suppose e is located at an i-point of an intersection u such that the invariant is satisfied, and the following move by the pursuers violates the invariant. Let the i-points of u be u_x, u_y, and u_z. We claim a pursuer can be within distance $1+2\delta$ of an i-point of at most one of x, y, and z, and break our analysis into two cases, either p lies within distance $1-2\delta$ of an i-point of u, or not.

In the first case, suppose without loss of generality p is within distance $1-2\delta$ of u_x, meaning it lies in the corridor from u to x. Then, as the invariant held before p moved necessarily $d(p,u_x) \geq 2\delta$. Further, as the i-points of u are 2δ apart, it is easy to see that $d(p,u_y) \geq 4\delta$, and $d(p,u_z) \geq 4\delta$. Thus, as $d(u_y,y_u) = 1-2\delta$ and $d(u_z,z_u) = 1-2\delta$ it follows that p is at least distance $1+2\delta$ from the i-points of y and z.

Consider the second case where p is further than $1-2\delta$ from the i-points of u and within $1+2\delta$ of i-points of two intersections in the set $\{x,y,z\}$. Without loss of generality suppose they are y and z. Then there exists i-points y_v and z_w such that $d(p,y_v) < 1+2\delta$ and $d(p,z_w) < 1+2\delta$. Consider the following cycle, $p, y_v, y_u, u_y, u_z, z_u, z_w, p$, which has length at most $(1+2\delta)+2\delta+(1-2\delta)+2\delta+(1-2\delta)+2\delta+(1+2\delta) = 4+6\delta$. This cycle then has length less than 5, as we can always construct P with an arbitrarily small δ. Further, as p is at least $1-2\delta$ from the i-points of u, the shortest paths from p to y_v and z_w to p cannot pass through a corridor adjacent to u without being longer than $1+2\delta$, thus this cycle surrounds one or more holes of P. However, G has no cycles of length three or four, thus the cycle in P then must have length five or more, and this is a contradiction.

Thus each pursuer is within distance $1+2\delta$ of an i-point of at most one intersection in the set $\{x,y,z\}$. Thus one of x_u, y_u, and z_u will satisfy the invariant and as they are all within distance one of the i-points of u, e can move to the one which satisfies the invariant. Thus, at each turn e can re-establish the invariant and indefinitely avoid capture.

□

7 Closing Remarks

In this paper, we proved that three pursuers are always sufficient to capture an evader in a polygonal environment of arbitrary complexity, under the assumption that pursuers have access to evader's location at all times. We also proved a matching lower bound, showing that three pursuers are also necessary in the worst-case. Traditionally, the papers on continuous space, visibility-based pursuit problem have focused on simply detecting the evader, and not on capturing it. One of our contributions is to isolate the *intrinsic* complexity of the capture from the associated complexity of detection or localization. In particular, while $\Theta(\sqrt{h} + \log n)$ pursuers are necessary (and also sufficient) for detection or localization of an (arbitrarily fast) evader in a n-vertex polygon with h holes [8], our result shows that full localization information allows capture with only *3* pursuers. On the other hand, it still remains an intriguing open problem whether $\Theta(\sqrt{h}+\log n)$ pursuers can *simultaneously* perform localization and capture. We leave that as a topic for future research.

8 Acknowledgements

The authors would like to thank Lijie Ren for helpful insights into the proof of Lemma 2.

A Capture in a Simply-Connected Polygon

Isler et al. [11] 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 squared distance from o increases by at least $1/n$. 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 13 (Capture in a simply connected polygon [11]). *A single pursuer can capture the evader in any simply-connected polygon P in $O(n \cdot diam(P)^2)$ moves.*

References

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

[2] D. Bhadauria and V. Isler. Capturing an evader in a polygonal environment with obstacles. In *International Joint Conference on Artificial Intelligence*, 2011.

[3] D. Bienstock and P. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12(2):239–245, 1991.

[4] T. H. Chung, G. A. Hollinger, and V. Isler. Search and pursuit-evasion in mobile robotics. *Auton. Robots*, 31:299–316, November 2011.

[5] F. V. Fomin, P. A. Golovach, and J. Kratochvíl. On tractability of cops and robbers game. In *TCS*, pages 171–185, 2008.

[6] A. S. Goldstein and E. M. Reingold. The complexity of pursuit on a graph. *Theoretical Computer Science*, 143(1):93 – 112, 1995.

[7] 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.

[8] L. J. Guibas, J.-C. Latombe, S. M. LaValle, D. Lin, and R. Motwani. Visibility-based pursuit-evasion in a polygonal environment. *IJCGA*, 9(5):471–494, 1999.

[9] B. Halpern. The robot and the rabbit–a pursuit problem. *The American Mathematical Monthly*, 76(2):140–145, 1969.

[10] J. Hershberger and S. Suri. "Vickrey pricing and shortest paths: What is an edge worth?". In *43th FOCS*, 2002.

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

[12] V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion with local visibility. *SIAM Journal on Discrete Mathematics*, 1:26–41, 2006.

[13] V. Isler and N. Karnad. The role of information in the cop-robber game. *TCS*, 399(3):179 – 190, 2008.

[14] K. Klein and S. Suri. Complete information pursuit evasion in polygonal environments. In *Proceedings of the 25th Conference on Artificial Intelligence (AAAI)*, pages 1120–1125, 2011.

[15] A. LaPaugh. Recontamination does not help to search a graph. *J. ACM*, 40(2):224–245, 1993.

[16] J. E. Littlewood. *Littlewood's Miscellany*. Cambridge University Press, 1986.

[17] E. Nardelli, G. Proietti, and P. Widmayer. Finding the most vital node of a shortest path. *TCS.*, 296(1):167–177, 2003.

[18] S.-M. Park, J.-H. Lee, and K.-Y Chwa. Visibility-based pursuit-evasion in a polygonal region by a searcher. In *ICALP*, pages 281–290. Springer-Verlag, 2001.

[19] T. D. Parsons. Pursuit-evasion in a graph. In Y. Alavi and D. R. Lick, editors, *Theory and Application of Graphs*, pages 426–441. Springer-Verlag, Berlin, 1976.

[20] S. Sachs, S. Rajko, and S. M. LaValle. Visibility-based pursuit-evasion in an unknown planar environment. *International Journal of Robotics Research*, 23(1):3–26, 2004.

[21] I. Suzuki and M. Yamashita. Searching for a mobile intruder in a polygonal region. *SIAM J. Comp.*, 21:863–888, 1992.

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

[23] J. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):pp. 712–716, 1971.