Catch Me If You Can: Pursuit and Capture in Polygonal Environments with Obstacles*

Kyle Klein and Subhash Suri

Department of Computer Science
University of California
Santa Barbara, CA 93106
{kyleklein, suri}@cs.ucsb.edu

Abstract

We resolve a several-years old open question in visibility-based pursuit evasion: how many pursuers are needed to capture an evader in an arbitrary polygonal environment with obstacles? The evader is assumed to be adversarial, moves with the same maximum speed as pursuers, and is "sensed" by a pursuer only when it lies in line-of-sight of that pursuer. The players move in discrete time steps, and the capture occurs when a pursuer reaches the position of the evader on its move. Our main result is that $O(\sqrt{h} + \log n)$ pursuers can always win the game with a deterministic search strategy in any polygon with n vertices and h obstacles (holes). In order to achieve this bound, however, we argue that the environment must satisfy a *minimum feature size* property, which essentially requires the minimum distance between any two vertices to be of the same order as the speed of the players. Without the minimum feature size assumption, we show that $\Omega(\sqrt{n/\log n})$ pursuers are needed in the worst-case even for *simply-connected* (hole-free) polygons of n vertices! This reveals an unexpected subtlety that seems to have been overlooked in previous work claiming that $O(\log n)$ pursuers can always win in simply-connected n-gons. Our lower bound also shows that *capturing* an evader is inherently more difficult than just "seeing" it because $O(\log n)$ pursuers are provably sufficient for line-of-sight detection *even against an arbitrarily fast* evader in simple n-gons.

Introduction

Pursuit evasion games arise in applications ranging from military strategy to trajectory tracking, search-and-rescue, monitoring, surveillance and so on (Alspach 2004; Chung, Hollinger, and Isler 2011; Fomin and Thilikos 2008). In the last few decades, they have spawned a significant body of research in graph searching, differential games, robotics, control theory and geometric algorithms (Alexander, Bishop, and Ghrist 2006; Bopardikar, Bullo, and Hespanha 2007; Isaacs 1965; LaPaugh 1993; Parsons 1976; Suzuki and Yamashita 1992). The general problem addressed in this paper relates to pursuit evasion in *continuous* space under a visibility-based model of sensing (Suzuki and Yamashita 1992).

*We gratefully acknowledge the support by National Science Foundation under the grant IIS-0904501.

In particular, we have an environment modeled as a polygon P, possibly containing obstacles. We use h to denote the number of obstacles, a mnemonic for "holes" in the polygon, and n to denote the total number of vertices in the environment, including the obstacles. A team of pursuers $\{p_1, p_2, \ldots, p_k\}$ is tasked to locate and catch a mobile evader e, and the fundamental question we seek to answer is this: *how many pursuers are always sufficient to catch the evader, no matter what strategy it follows?* Specifically, as a function of n and h, what is the smallest number k of pursuers that guarantees a win for the pursuers in any polygon of n vertices and h holes.

The history of pursuit-evasion games in geometric environments is long, and can be traced to the celebrated "Lion-and-Man" problem, attributed to Rado in 1930s: if a man and a lion are confined to a closed arena, and both have equal maximum speeds, can the lion catch the man? Surprisingly, the man can evade the lion indefinitely as shown by Besicovitch (Littlewood 1986)—the lion fails to reach the man in any finite time although it can get arbitrarily close to him. If we allow the capture to occur when the lion gets within distance ε of the man, then the lion wins in $O(r \log \frac{r}{\varepsilon})$ time (Alonso 1992), where r is the radius of the circle, and the maximum movement speed is one. An interested reader may consult (Bopardikar, Bullo, and Hespanha 2008; Chung, Hollinger, and Isler 2011; Kopparty and Ravishankar 2005; Sgall 2001) for several other variations of the problem.

The distinguishing feature of our work is the *complexity of the geometric environment*, in particular, the presence of obstacles that impede visibility and produce many different homotopy classes of escape paths for the evader. Indeed, even for the lion-and-man game, the only algorithmic work exploring the effect of obstacles is by Karnad and Isler (Karnad and Isler 2009), which considers the presence of a *single circular obstacle*. Because of its adversarial nature, pursuit evasion is also studied as a form of differential games and solved using the Hamilton-Jacobi-Isaacs equation. Unfortunately, the resulting system of differential equations is intractable for all but the simplest of the environments, and unsuited for the complex, multiply-connected environments we consider.

The most relevant work to our research is the paper by Guibas et al. (Guibas et al. 1999), which introduced a formal framework and analysis of visibility-based pursuit in complex polygonal environments. In order to make the problem tractable, however, Guibas et al. make one crucial simplifying assumption: *evader loses if it is "seen" by any pursuer.* That is, the pursuers need to only detect the presence of the evader, and not physically catch it. With this weaker requirement of "capture," Guibas et al. manage to prove several interesting combinatorial bounds, including that $\Theta(\log n)$ pursuers in a simply-connected polygon, and $\Theta(\sqrt{h} + \log n)$ pursuers in a polygon with h holes (obstacles), are always sufficient and sometimes necessary. In fact, these bounds hold even if the evader can move arbitrarily faster than the pursuers.

In the intervening twelve or so years, there has been little progress on extending these "detection" of evader bounds to physical "capture" of the evader. In fact, there are only a handful of small results to speak of. First, Isler et al. (Isler, Kannan, and Khanna 2005) show that in simply-connected polygons, two pursuers can capture the evader in *expected* polynomial time using a randomized strategy. A deterministic version of their strategy works with $O(\log n)$ pursuers. Recently, and almost simultaneously, two groups (Bhadauria and Isler 2011) and (Klein and Suri 2011) proved that *if the location of the evader is always known to the pursuers*, e.g., using an ubiquitous camera network, then 3 pursuers are enough to win the game. Without these extreme conditions of unfair advantage to the pursuers, no non-trivial upper bound on the number of pursuers necessary to win the game is known. The main result of this paper is to resolve this question, and in the process uncover a mathematical subtlety that has been overlooked by previous work. In particular, our paper makes the following contributions.

Statement of Results

We first prove a general lower bound of $\Omega(\sqrt{n/\log n})$ for the number of pursuers needed in the worst-case to catch an equally fast evader in *simply-connected* (hole-free) polygons of n vertices. This lower bound reveals an inconsistency with the existing *upper bound* of $O(\log n)$ pursuers for the same problem (Isler, Kannan, and Khanna 2005). In trying to reconcile this inconsistency, we discovered that the previous upper bound makes some implicit assumptions about the geometry of the environment, in particular the speed of the players relative to the size of the environment. Our lower bound shows that one needs to be careful about features of the environment and *not just the equality of the speed of the evader and the pursuers.*

We then show that a *minimum feature size* property of the environment is sufficient to yield significantly better and natural upper bounds for the capture problem. The minimum feature size of a polygonal environment is the *minimum* distance between any two vertices (using the shortest path metric in free space). We normalize the maximum speed of the players to one, and require that the minimum feature size of the environment to be at least one. We feel that this condition naturally occurs in physical systems, and its violation is the source of the $\Omega(\sqrt{n/\log n})$ lower bound mentioned earlier. With this assumption, we show that $O(\sqrt{h} + \log n)$ pursuers are always sufficient to catch the evader in a multiply-connected polygon of n vertices and h obstacles (holes). When the polygon is simply-connected (hole-free), this yields an $O(\log n)$ bound for the number of pursuers. The pursuers' winning strategy is deterministic, and succeeds in polynomial time.

Preliminaries

Our pursuit evasion problem is set in a two-dimensional closed polygonal environment P, with a total of n vertices and h holes (where n includes the vertices of the holes). A team of pursuers $\{p_1, p_2, \ldots, p_k\}$ is tasked to locate and catch an evader e. The evader and the pursuers have the same maximum speed, which we assume to be one without loss of generality. Abusing the notation slightly, we also denote the current positions of the players as p_i and e. The players' sensing model is based on line-of-sight visibility: a pursuer can see the evader only if the latter is in the pursuer's line of sight. That is, a pursuer p sees the evader e only if the line segment (p, e) does not intersect the boundary of the environment. The pursuers are assumed to operate in a global communication model so that they can coordinate their moves, and share their knowledge of the evader's location.

The players take alternate turns, where all the pursuers can move simultaneously on their turn. In each turn, a player can move to any position that is within distance one of its current location, under the shortest path metric, in the *free space* (the region of the polygonal environment not occupied by holes). We assume that all the players know the environment, and the pursuers do not know the strategies or the future moves of the evader, although the evader may have complete information about pursuers' strategies. The pursuers *win* if after a finite amount of time, some pursuer becomes collocated with the evader. The evader wins if it can evade indefinitely. The fundamental question we seek to answer is this: *what is the minimum value of k, the number of pursuers, that guarantees a win for the pursuers?*

We begin by establishing a lower bound that shows the difficulty of capturing the evader in the general case, and then introduce a natural geometric condition for the environment, called the *minimum feature size*, which allows us to prove significantly better upper bounds.

A General Lower Bound

Without any restrictions on the environment or the speed of the players, except that pursuers and evader have the same maximum speed, we prove the following lower bound.

Theorem 1. *There exist simply-connected (hole-free) polygons with n vertices that require at least $\Omega(\sqrt{n/\log n})$ pursuers to catch an equally fast evader.*

Proof. We give an explicit construction that forces the claimed number of pursuers. Our polygons correspond to complete binary trees in which an n-node tree is mapped to a simple polygon of at most $12n$ vertices. Each edge of the tree T maps to a "corridor with a bend," which we call an *edge* *corridor*, and each vertex maps to a *vertex corridor* that interconnects the edge corridors of its incident edges. Figure 1 shows the construction for a binary trees with 4 leaves. We now show that this polygon can always be constructed such that the *vertex corridors are no longer than edge corridors*, thereby ensuring that the length of a shortest path between any two points in the polygon is at most $4\log n$—such a path visits at most $2\log n$ edge and vertex corridors each. We initially construct the polygon so that each edge corridor has length 1, while the vertex corridors get progressively longer at higher levels of the tree. Since the number of nodes at each level is halved, the length of the vertex corridor doubles. Because the tree has height $\log n$, the longest vertex corridor (at the root) level has length $\Theta(n)$, and so the polygon can be embedded on a $O(n) \times O(\log n)$ size grid. Next, to achieve the property that vertex corridors are shorter than edge corridors, we apply a *scaling transform* along the y-axis so that all edge corridors are stretched by a factor L, where L is the length of the longest vertex corridor before the scaling. Finally, we scale the entire polygon down to make the edge corridors one unit long, thus ensuring that the shortest path between any two points of the polygon has length at most $4\log n$.

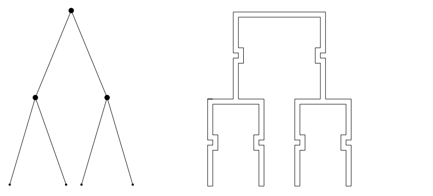


Figure 1: (a) A binary tree; (b) the corresponding polygon.

We now argue that this construction requires $\Omega(\sqrt{n/\log n})$ pursuers to catch the evader, by presenting an evader strategy that avoids its capture with fewer pursuers. Suppose that a certain number of pursuers, say p, each with maximum speed $4\log n$, are sufficient to win against the evader, who also has the maximum speed of $4\log n$. Let us first consider the line-of-sight visibility of any pursuer. A pursuer in a vertex corridor can only see within that corridor plus at most three edge corridors incident to that vertex corridor. A pursuer in an edge corridor can see that corridor plus at most one incident vertex corridor. Therefore, the p pursuers collectively have visibility of at most p vertex corridors and their $3p$ adjacent edge corridors. An evader located in any other corridor is invisible to the pursuers.

We now reason about evader's strategy by considering how the pursuers' positions partition the tree T. Consider deleting from the tree T the p vertices and their $3p$ adjacent edges corresponding to the corridors visible to the pursuers. This partitions the tree into at most $2p+1$ connected components—each deletion splits one component into three, thus increasing the component count by 2. By the pigeonhole principle, the largest component (call it C) must have at least $\frac{n-p}{2p+1}$ nodes, and at least $\lfloor\frac{n-p}{2(2p+1)}\rfloor$ non-leaf nodes (corridors); the bound on non-leaf node holds because we have a binary tree. The evader's strategy is to move into this largest component C on its next move. Since C is pursuer-free, the entire component must be searched to locate and capture the evader. Each edge corridor has length one, any pursuer moving at maximum speed can search at most $4\log n + 1 < 5\log n$ nodes (the pursuer may be closer than one to the first corridor searched, so we give that one for free), and so the p pursuers collectively can search at most $5p\log n$ nodes of C.[1] Thus, for the pursuers to capture the evader on their move, the following inequality must be satisfied:

$$5p{\cdot}\log n \;\geq\; \lfloor\frac{n-p}{2(2p+1)}\rfloor - 1 \;>\; \frac{n-p}{5p} \;>\; \frac{n-n/2}{5p} \;=\; \frac{n}{10p},$$

which leads to $p \geq (\frac{n}{50\log n})^{1/2}$. (We have made no attempt to optimize the constant factor in our proof, and instead focused on a quick derivation of the asymptotic bound.) Since the polygon corresponding to a n-node tree has at most $12n$ vertices, we conclude that in the worst-case capturing an evader in a simply-connected n-gon requires $\Omega(\sqrt{n/\log n})$ pursuers. $\square$

Remark: Our lower bound seems to contradict a result of (Isler, Kannan, and Khanna 2005) that $O(\log n)$ pursuers can always capture an equally fast evader. (Strictly speaking, their result claims win for just *two* pursuers using a randomized strategy. However, the randomization is used only for *locating* the evader, and so their solution can be made deterministic by performing localization using $O(\log n)$ pursuers (Guibas et al. 1999).) The source of the contradiction is that Isler et al. *implicitly* assume that the environment is *much larger than the speed of the players*. In "practice" this is an eminently reasonable assumption—their choice of speed or "step size" is an attempt to discretely approximate the "continuous motion" of players. However, mathematically, this is unsatisfactory because precisely what properties of the environment are needed is not made clear. In concrete terms, their "proof" that each vertex of the polygon is the "blocking vertex" *at most once* is not valid in general, with the polygon of our lower bound construction (Fig. 1) being a particular counter-example.

In the remainder of the paper, we show that a simple geometric condition on the environment, which we call the *minimum feature size* condition, allows us to circumvent the lower bound of Theorem 1. In particular, we show that if the environment is a simply-connected n-gon with minimum feature size, then $O(\log n)$ pursuers can always win the game. Our main result, however, is to settle the complexity of pursuit evasion in polygonal environments with holes, where no non-trivial bound was previously known.

Upper Bounds For Environments with Minimum Feature Size

We begin with a definition of the minimum feature size.

[1]In fact, our estimate is conservative because the edges corresponding to leaves must be searched too—however, it does not change the asymptotics of the lower bound.

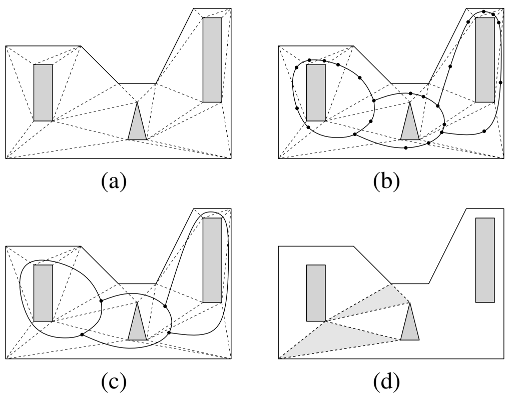


Figure 2: (a) A triangulation; (b) its dual graph; (c) the contracted dual; (d) illustration of the recursive strategy—two separating triangles (shaded) break the environment into 3 connected regions, each with at most 1 obstacle.

Definition 1. Minimum Feature Size: *The minimum feature size of a (multiply-connected) polygon P is the minimum distance between any two vertices, where the distance is measured by the shortest path within the polygon.*

We will assume that the minimum feature size of the environment is *lower bounded* by the maximum speed of the players: i.e., the environment has minimum feature size of at least one. One can check that the polygon used in our lower bound (Figure 1) violates the minimum feature size: the players' maximum speed is $4 \log n$ while the edges forming the corridors or bends are closer than even 1.

The High Level Pursuit Strategy

We begin with a high-level description of the pursuers' winning strategy. For ease of reference, we will split the pursuers into three categories: *guards*, *searchers*, and a *catcher*. The search is based on a divide-and-conquer strategy that recursively partitions the environment, installing some pursuers to guard the "separating triangles," until the search reaches simply-connected regions, at which point the searchers locate the evader and the catcher captures it. In particular, the strategy has the following main steps.

Algorithm RecursivePursuit

1. Compute a triangulation of the environment P.
2. Contract the dual of triangulation to an $O(h)$ size graph G.
3. Use the Planar Separator Theorem on G to identify $O(\sqrt{h})$ triangles that partition P into two sub-polygons, each containing at most $2h/3$ holes.
4. Guard each "separating triangle" with a constant number of pursuers so that the evader cannot move across the partition without being captured.
5. Recursively search one side of partition, then the other.
6. The recursion stops when the sub-polygon has no holes. Then, use our algorithm **SimplePursuit** to search for, and capture, the evader.

The remainder of this section describes the details of this strategy, but let us briefly discuss the complexity bound for this strategy. A polygon with n vertices, and any number of holes, can be triangulated in $O(n \log n)$ time (de Berg et al. 1997). We reduce this $O(n)$ size triangulation graph to an $O(h)$ size planar graph by repeatedly *contracting* each graph vertex of degree 2, and deleting every vertex of degree one. (See Figure 2.) In the end, each surviving vertex has degree 3, and Euler's formula for planar graph implies that G has h faces, at most $2h - 2$ vertices, and at most $3h - 3$ edges.

We use this graph G to recursively divide the environment into two parts, so that each part has a constant fraction of the holes. In the base case we reach simply-connected sub-polygons of P. Specifically, we use the following well-known Planar Separator Theorem (Lipton and Tarjan 1977; Djidjev 1982).

Lemma 1. *(Djidjev 1982) Every planar graph $G = (V, E)$ on n vertices admits a partition of the vertices into three sets A, S, and B, such that neither A nor B has more than $2n/3$ vertices, S has at most $\sqrt{6n}$ vertices, and there are no edges with one endpoint in A and the other endpoint in B. The set of vertices S is called a* separator *of G.*

Using this separator theorem, we can partition G into two parts, each containing at most $2h/3$ nodes, by deleting a set of $O(\sqrt{h})$ nodes. In the geometric space, this separator corresponds to a set of $O(\sqrt{h})$ triangles that divide the environment P into two parts, each containing at most $2h/3$ holes. (See Figure 2(d).)

Our main technical result, which is described in the remainder of this section, is to show (i) how to guard each of the separating triangles with a constant number of pursuers to prevent the evader from escaping across the partition, and (ii) how to search and capture the evader in the terminal case of a hole-free sub-polygon. With those pieces in place, it is now easy to see that the number of pursuers needed is $O(\sqrt{h} + \log n)$, as follows. We will show that the algorithm SimplePursuit needs only $O(\log n)$ guards to search, and one additional pursuer to capture the evader when it is found. Altogether, these $O(\log n)$ pursuers are *reused* throughout the divide and conquer strategy, so they form only an additive term. The primary demand for the pursuers is in the form of *guards*, which are placed at the separating triangles throughout the divide-and-conquer. Their count has the following recurrence: $T(h) = T(2h/3) + c\sqrt{h}$, where c is a constant. This well-known recurrence solves to $T(h) = O(\sqrt{h})$, easily proved by induction, and so the total number of pursuers used by the algorithm is $O(\sqrt{h} + \log n)$.

Geometry of Guarding and Capture: Preliminaries

We begin with some technical preliminaries that form the basis of our geometric pursuit strategies and analysis. Throughout, we use the notation $d(a, b)$ for the free-space distance between two positions a and b. That is, given two points a and b in the environment, $d(a, b)$ is the length of the shortest obstacle-avoiding path between them. By normalization, we assume that the maximum speed of the players is one—that is, a player can move from position a to b in the environment in one move only if $d(a, b) \leq 1$. The following lemma gives a technical condition under which a pursuer is able to capture the evader.

Lemma 2. *Consider a free space line segment ab containing a pursuer p. If a move by the evader, from position e to e', crosses the segment ab at a point x that is closer to p than to e, then p can capture the evader on its next move.*

Proof. We first observe that the condition of the crossing point x implies that the evader's final position is within distance one from p. This follows from the triangle inequality: $d(p, e') \leq d(p, x) + d(x, e') \leq d(e, e') \leq 1$. If e' is visible to p, then the capture is clearly trivial, so let's assume that the final position e' is not visible to p.

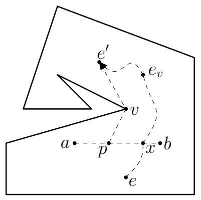


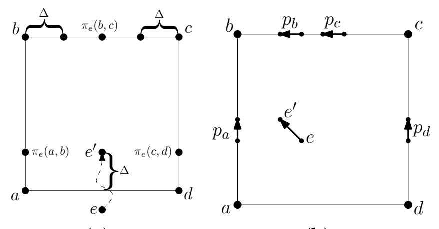
Figure 3: The proof of Lemma 2.

Let e_v be the last position during the move when the evader is visible to p, which is necessarily further along than x on the evader's path. Using the triangle inequality again, we conclude that $d(p, e_v) \leq d(e, e_v)$. The line segment pe_v must contain a vertex of the environment, call it v, blocking p's visibility past the point e_v (see Figure 3). We claim that the shortest path homotopic to (p, x, e_v, e') is (p, v, e'), that is, it consists of a single vertex v. This follows from our minimum feature size assumption. Since the path (p, x, e_v, e') has length at most 1, the shortest path of the same homotopy also has length at most one, and the minimum feature size forbids two vertices with shortest path distance less than one. Thus, v is visible from both p and e', and $d(p, v) + d(v, e') \leq 1$. The pursuer p, therefore, can capture by first moving to v and then to e' in a single move. This completes the proof. □

The key idea in the pursuers' strategy is to guard *separating triangles*, as outlined earlier. Unfortunately, triangles can be long and skinny, which makes guarding them against an adversarial evader difficult. We, therefore, employ a more complex and *indirect* strategy, which involves *covering* the triangles with easier-to-guard shapes, namely, squares. As a first step towards this goal, we define a *trap* to be an area within the environment, guarded by a constant number of pursuers, that cannot be "crossed" by the evader without being captured. Our next lemma shows that *squares* are feasible traps. In order to explain the main idea cleanly, we first focus on a square region in isolation, without worrying about the vertices or edges of the polygonal environment P. Define a *crossing path* as a *sequence of moves* by the evader in which it enters the square from one side and exits through a different side.[2]

[2]If the evader "peeks" in through a side and later retracts through the same side, then that does not qualify as a crossing path.

Lemma 3. *Consider a square-shaped region R lying entirely in the free space of the environment P. Four guards, initially placed one at each corner, can guarantee that any crossing path of the evader leads to its capture.*



(a) (b)

Figure 4: Illustrating the proof of Lemma 3.

Proof. The proof is elementary but technical, highlighting the subtle geometry of continuous space adversarial pursuit. Due to space limitation, we describe the main idea of the proof, omitting technical details to the full version of the paper. Without loss of generality, assume that the evader enters the square R from the bottom edge, namely ad. Let p_a, p_b, p_c, p_d, respectively, denote the guards originally placed at the four corners a, b, c, d. The pursuers p_a and p_d will guard the edges ab and cd, respectively, and the remaining two pursuers guard the edge bc. See Figure 4(a). To guard the edge ab, the pursuer p_a always moves to a defensive position on its turn, which we call the *projection* of the evader on ab. The projection, denoted $\pi_e(ab)$, is the point on ab that is closest to e', namely, the foot of the perpendicular from e' to ab. By maintaining this position, p_a always satisfies the condition of Lemma 2, ensuring e's capture if it were to cross ab. The same argument holds for p_d and the edge cd. The remaining case of bc turns out to be more complicated because neither p_b nor p_c may be near the projection point if the square's side length bc is much larger than player's speed. We show that by coordinating their moves carefully to track the evader, the two pursuers p_b, p_c can also prevent crossing of bc. □

Using the feasibility of a square-shaped trap, we next complete our goal of guarding the triangles by the following two steps: (1) cover each side of the triangle with a constant number of square traps, and (2) carefully choose a particular triangulation that permits such a covering by squares *even in the presence of obstacle boundaries.* This is the topic of the next subsection.

Triangle Covering by Squares and Constrained Delaunay Triangulation

We show that each side of the triangle can be covered with either one square, or two squares whose convex hull is a trap in the shape of a pentagon. We call such a trap placement a *cover*. The side of the triangle is either itself a side of the cover or its diagonal. This trap has the property that unless the evader re-crosses the same side of the triangle, any attempt to exit the trap will result in a crossing path, and the evader's capture.

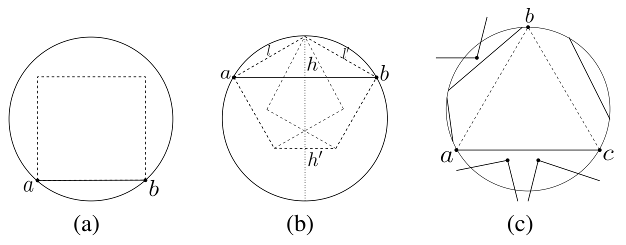


Figure 5: Illustrating the proofs of Lemmas 4 and 5.

Lemma 4. *Consider a triangle $\triangle \equiv \triangle abc$ whose circumcircle does not contain any vertices or edges of the polygonal environment P. Then, a side of $\triangle$ can be covered using at most two* free space *squares.*

Proof. We denote the circumcircle of the triangle by C, and let r be its radius. Without loss of generality, we explain how to cover the side ab of $\triangle$. In a circle of radius r, the largest inscribed square has side length $r\sqrt{2}$. Thus, if $|ab| \leq r\sqrt{2}$, we can erect a square on ab as its side lying entirely within C. This square is the cover of ab; see Figure 5(a).

On the other hand, if $|ab| > r\sqrt{2}$, we cover ab by combining two squares, as shown in Figure 5(b). Consider the perpendicular bisector of ab, and the two segments h and h' into which ab divides the bisector. Since the bisector cannot be longer than the diameter of C, either h or h' must have length at most r. Without loss of generality, suppose that h is the shorter one. Then, the two right triangles created by ab and h have hypotenuses, call them l and l', of length at most $r\sqrt{2}$, and so we can place two squares, one with l as a side and one with l' as a side. The convex hull of these two squares (which is itself a square when ab is the diameter, and otherwise a pentagon) is a trap that covers ab. ☐

In general, a polygonal environment (even one without holes) cannot be triangulated to achieve the condition required by the preceding lemma, namely, triangles whose circumcircles do not intersect obstacle boundaries. We, therefore, need to settle for something weaker, but sufficient for our need. We need the concept of a *Constrained Delaunay Triangulation* (CDT) from computational geometry (Shewchuk 1999). Specifically, a constrained Delaunay triangulation of a polygon (with holes) is a triangulation in which (1) every edge of the polygon appears as an edge of the triangulation, (2) the circumcircle of any triangle does not contain any *visible* vertex of the polygon in its *interior*. The visibility is defined as line-of-sight. Figure 5(c) shows an example. We show in the following lemma that the exclusion of "visible vertices" from the circumcircle of a triangle is sufficient to cover each side of the triangles.

Lemma 5. *Consider a triangle $\triangle \equiv \triangle abc$ whose circumcircle does not contain any* visible vertices *of the polygonal environment P. Then, any side of $\triangle$ can be covered using at most* 16 *guards.*

Proof. The proof basically shows that because of the CDT property, the only vertices that lie in the circumcircle C must be *invisible* from $\triangle$, and therefore we do not need to guard the portions of a covering square that are invisible. Due to page limitation, we omit the mostly technical, but conceptually straightforward, details of the proof. ☐

The Algorithm SimplePursuit

Capturing entails two distinct tasks: localizing the evader's position, and physically arriving at that location. The lower bound of our Theorem 1 shows that with sufficient speed, the evader can continuously nullify the pursuers localization efforts. However, if the pursuers maintain full visibility of the evader, then the situation is essentially equivalent to the un-obstructed game of lion and man. This was formalized by Isler et al, and we cite their result.

Lemma 6. *(Isler, Kannan, and Khanna 2005) In an n vertex hole-free polygon P, if a pursuer p knows the location of an evader at all times, then it can capture e in $O(n \cdot \mathrm{diam}(P)^2)$ moves.*

Thus the key problem for us is to show that pursuers can maintain full visibility of the evader *when we invoke the final step of SimplePursuit*. The following algorithm describes the main steps of SimplePursuit, called on a simply-connected environment P.

Algorithm SimplePursuit

1. Compute a constrained Delaunay triangulation (CDT) of the environment P.
2. Choose a node of the dual graph (separating triangle) to partition P into two or more sub-polygons, each containing at most $n/2$ triangles.
3. Place at most 6 squares, with at most 8 guards each, to guard the "separating triangle" so that the evader cannot move across the partition without being captured.
4. Locate the evader, then recursively pursue the evader in the sub-polygon containing it.
5. The recursion stops when e lies in a triangle surrounded by guarded triangles, at which point capture it using Lemma 6.

We now prove the following result.

Theorem 2. *Given a simply-connected polygon P of n vertices with minimum feature size, $O(\log n)$ pursuers can always capture the evader in $O(n \cdot \mathrm{diam}(P) + \mathrm{diam}(P)^2)$ worst-case number of moves.*

Proof. This polygon-halving recursive strategy requires a total of $\log n$ triangles to be guarded, each requiring a constant number of guards. The terminal case of the search occurs when the evader is confined to a single triangle surrounded by three guarded triangles. Each trap is *convex* (square or pentagon), and the guards can see the entire trap they guard as well as the triangle they cover. Further, the number of vertices in the terminal sub-polygon is a constant, its diameter is at most $2 \cdot \mathrm{diam}(P)$, and so by Lemma 6 the capture takes $O(\mathrm{diam}(P)^2)$ number of moves. Throughout the divide-and-conquer, we have $\log n$ localization searches, but the number of vertices in the sub-polygons shrink by half each time, giving a telescopic sum of $\sum_{i=1}^{\log n} \frac{c \cdot n}{2^i} \mathrm{diam}(P) = O(n \cdot \mathrm{diam}(P))$. The theorem follows. ☐

Analysis of RecursivePursuit

We now have all the machinery in place to complete the proof that the algorithm **RecursivePursuit** successfully captures the evader in a polygonal environment with obstacles. We will need a technical lemma that extends the localization algorithm of Guibas et al. (Guibas et al. 1999) to regions within multiply-connected environments.[3] (Due to space limitation, we omit its technical, but simple, proof.)

Lemma 7. *Let P be a multiply-connected polygonal environment with n vertices, and suppose the evader is confined to remain inside a simply-connected polygonal subspace S of P, bound by k vertices. Then, using $O(\log k)$ pursuers, we can locate the evader within S in $O(k \cdot \operatorname{diam}(P))$ number of moves.*

We are now ready to state and prove the main result of our paper.

Theorem 3. *Let P be a polygonal environment with n vertices and h disjoint obstacles (holes), satisfying the minimum feature size. Then, $O(\sqrt{h} + \log n)$ pursuers are always sufficient to simultaneously locate and capture an equally fast evader in $O(n \cdot \operatorname{diam}(P) + \operatorname{diam}(P)^2)$ number of moves.*

Proof. The recursive algorithm searches a total of h hole-free regions. These regions are disjoint, except sharing of the separating triangles' traps. If the ith hole-free region has n_i vertices, where $\sum_i n_i = O(n)$, then by Lemma 7, searching it takes $O(n_i \cdot \operatorname{diam}(P))$ moves, which sum to $O(n \cdot \operatorname{diam}(P))$. The total time needed to position guards at all the separating triangles is $O(h \cdot \operatorname{diam}(P))$, because each pursuer needs to move at most $O(\operatorname{diam}(P))$ distance. When the evader is finally confined to a hole-free sub-polygon, the capture is guaranteed to occur in $O(n \cdot \operatorname{diam}(P) + \operatorname{diam}(P)^2)$ number of moves as shown by Theorem 2. This completes the proof. □

Closing Remarks

We have shown that $O(\sqrt{h} + \log n)$ pursuers can always capture an equally fast evader in any polygon with n vertices and h obstacles (holes). This matches the best bound known for just *detecting* an evader (a simpler problem) in the visibility-based pursuit. Our research also revealed an unexpected subtlety of the capture problem, relating the players' speed to the *minimum feature size* of the environment, which had not been realized in the previous work. Further, we show that capturing the evader is *provably* harder than detecting because capturing without the minimum feature size requires $\Omega(\sqrt{n/\log n})$ pursuers (in simply-connected polygons) while detection can be achieved with just $O(\log n)$ pursuers, even against an arbitrarily fast evader (Guibas et al. 1999).

[3]The *technical* claim of this lemma is that the localization can be accomplished in time proportional to the diameter of the *original environment*, namely, $\operatorname{diam}(P)$, *even if the subspace S can have significantly larger diameter*. Though counter-intuitive, in a multiply-connected polygon, a sub-polygon can have a larger diameter than the original polygon.

References

Alexander, S.; Bishop, R.; and Ghrist, R. 2006. Pursuit and evasion in non-convex domains of arbitrary dimensions. In *Proceedings of Robotics: Science and Systems*.

Alonso, L. 1992. "lion and man": Upper and lower bounds. *ORSA Journal on Computing* 4(4):447 – 457.

Alspach, B. 2004. Searching and sweeping graphs: A brief survey. *Le Matematiche* 59(1,2):5–37.

Bhadauria, D., and Isler, V. 2011. Capturing an evader in a polygonal environment with obstacles. In *Proc. of the Joint Conf. on Artificial Intelligence (IJCAI)*, 2054–2059.

Bopardikar, S. D.; Bullo, F.; and Hespanha, J. 2007. A cooperative homicidal chauffeur game. In *46th IEEE Conference on Decision and Control*, 4857 –4862.

Bopardikar, S. D.; Bullo, F.; and Hespanha, J. 2008. on discrete-time pursuit-evasion games with sensing limitations. *IEEE Transactions on Robotics* 24(6):1429–1439.

Chung, T. H.; Hollinger, G. A.; and Isler, V. 2011. Search and pursuit-evasion in mobile robotics. *Auto. Robots* 1–18.

de Berg, M.; van Krefeld, M.; Overmars, M.; and Schwarzkopf, O. 1997. *Computational Geometry: Algorithms and Applications*. Springer-Verlag.

Djidjev, H. N. 1982. On the problem of partitioning planar graphs. *SIAM J. on Alg. & Discrete Methods* 3(2):229–240.

Fomin, F. V., and Thilikos, D. M. 2008. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.* 399:236–245.

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

Isaacs, R. 1965. *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*.

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

Karnad, N., and Isler, V. 2009. Lion and man game in the presence of a circular obstacle. IROS'09, 5045–5050.

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

Kopparty, S., and Ravishankar, C. V. 2005. A framework for pursuit evasion games in rn. *Inf. Process. Lett.* 96:114–122.

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

Lipton, R. J., and Tarjan, R. E. 1977. Applications of a planar separator theorem. 162 –170.

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

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

Sgall, J. 2001. Solution of david gale's lion and man problem. *Theor. Comput. Sci.* 259(1-2):663–670.

Shewchuk, J. R. 1999. Lecture notes on delaunay mesh generation. `http://www.cs.berkeley.edu/~jrs/meshpapers/delnotes99.ps.gz`.

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