Computational Geometry 48 (2015) 205–220

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Capture bounds for visibility-based pursuit evasion [☆]

Kyle Klein [*], Subhash Suri

University of California, Santa Barbara, CA 93106, USA

ARTICLE INFO

Article history:
Received 7 February 2014
Accepted 30 September 2014
Available online 5 October 2014

Keywords:
Constrained Delaunay triangulation
Lion-and-man
Upper bound
Lower bound

ABSTRACT

We investigate the following problem in the visibility-based discrete-time model of pursuit evasion in the plane: *how many pursuers are needed to capture an evader in a polygonal environment with obstacles under the minimalist assumption that pursuers and the evader have the same maximum speed?* When the environment is a simply-connected (hole-free) polygon of n vertices, we show that $\Theta(n^{1/2})$ pursuers are both necessary and sufficient in the worst-case. When the environment is a polygon with holes, we prove a lower bound of $\Omega(n^{2/3})$ and an upper bound of $O(n^{5/6})$ for the number of pursuers that are needed in the worst-case, where n is the total number of vertices including the hole boundaries. More precisely, if the polygon contains h holes, our upper bound is $O(n^{1/2}h^{1/4})$, for $h \leq n^{2/3}$, and $O(n^{1/3}h^{1/2})$ otherwise. We then show that with additional assumptions these bounds can be drastically improved. Namely, if the players' movement speed is small compared to the "feature size" of the environment, we give a deterministic algorithm with a worst-case upper bound of $O(\log n)$ pursuers for simply-connected n-gons and $O(\sqrt{h} + \log n)$ for multiply-connected polygons with h holes. Further, if the pursuers are allowed to randomize their strategy, regardless of the players' movement speed, we show that $O(1)$ pursuers can capture the evader in a simply connected n-gon and $O(\sqrt{h})$ when there are h holes with high probability.

1. Introduction

In visibility-based pursuit-evasion, a team of *pursuers* is tasked with locating and capturing an adversary, called the *evader*, within a polygonal environment. The pursuers are equipped with cameras, able to maintain omni-directional line-of-sight visibility, and must plan and coordinate their moves until some pursuer can reach the same location as the evader. The problem is motivated by applications in robotics, and has drawn a significant interest since it was introduced by Suzuki and Yamashita [24], although much of the prior work has focused on the simpler problem of *evader detection*, where the pursuers win as soon as the evader is "seen" by some pursuer [10–12,20,25].

In this paper, we consider the complexity of physically capturing the evader within this visibility-based framework of pursuit-evasion. For our main results, we make only the minimally necessary assumption that all players (pursuers and evader) have equal maximum speed, *which is normalized to one* by appropriate scaling of the environment. On its turn, each

[☆] This research was supported in part by the National Science Foundation Grants IIS-0904501, CNS-1035917, CCF-1161495, and the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1144085. A preliminary version of this paper was presented at the 29th Annual Symposium on Computational Geometry (SoCG '13) [16]. Additionally, this work includes results presented at the 26th Annual Conference on Artificial Intelligence (AAAI '12) [15].

[*] Corresponding author.
E-mail addresses: kyleklein@cs.ucsb.edu (K. Klein), suri@cs.ucsb.edu (S. Suri).

http://dx.doi.org/10.1016/j.comgeo.2014.10.002
0925-7721/© 2014 Elsevier B.V. All rights reserved.

player can move along a path to any position within distance one of its current location, where the distance is measured using the shortest path (geodesic) distance avoiding the obstacles in the environment. The game is played in the discrete, alternating turn model: the evader moves first, and then the pursuers. (One practical reason for adopting the discrete time model is that the differential equations modeling the continuous time game are intractable for all but the simplest of environments.)

Our first result gives a tight bound of $\Theta(n^{1/2})$ for the number of pursuers needed to capture the evader when the environment is a simply-connected (hole-free) polygon of n vertices. Generalizing this result, we show that at least $\Omega(n^{2/3})$ pursuers are needed for capture in polygons with holes. Complementing this lower bound, we prove an upper bound of $O(n^{1/2}h^{1/4})$, for $h \le n^{2/3}$, and $O(n^{1/3}h^{1/2})$ otherwise, where h is the number of holes in the polygon. More simply, the upper bound is $O(n^{5/6})$. We then show with additional assumptions these bounds can be drastically improved. Namely, if the players' movement speed is small compared to the "features size" of the environment, we give a deterministic algorithm with a worst-case upper bound of $O(\log n)$ pursuers for simply-connected n-gons and $O(\sqrt{h} + \log n)$ for multiply-connected polygons with h holes. Further, if the pursuers are allowed to randomize their strategy, regardless of the players' movement speed, we show that $O(1)$ pursuers can capture the evader in a simply connected n-gon and $O(\sqrt{h})$ when there are h holes with high probability.

Related work 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 [18]—the lion fails to reach the man in any finite time although it can get arbitrarily close to him [2]. Extending this result to environments with obstacles, however, has proved difficult, and the only relevant result seems to be a recent work of Karnad and Isler [13] that deals with a single circular obstacle! 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 study in this paper. An interested reader may consult [5,7,17,22] for a general survey for many variations of the lion-and-man problem. There also exists a substantial body of work on graph searching and pursuit-evasion in graphs [1,19,21], but our focus is on geometric environments.

The most relevant work to our research is the paper by Guibas et al. [11], 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: *the 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 $\Theta(h^{1/2} + \log n)$ pursuers in a polygon with h holes (obstacles), are always sufficient and sometimes necessary. Further work by Isler et al. [12] provided a randomized algorithm for the pursuers in which a single pursuer can locate the evader with high probability in a simply connected polygon, and $O(\sqrt{h})$ pursuers when there are h holes. Many other variations of the visibility-based pursuit-evasion have been studied over the years, including detection by a chain of guards [9] or guards with a limited field of view [10], but their focus remains detection, not capture.

Indeed, until recently, there had been little progress on extending these detection-of-evader bounds to physical capture of the evader. About a year ago, independently and simultaneously, two groups [3] and [14] 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.[1] In a sense, this research suggests that "localization" of the evader is the more difficult part of the pursuit evasion, and the evader's power comes from its ability to "disappear" from the collective sights of all the pursuers.

The results of the current paper seek to compare the bounds on the number of pursuers required to locate or capture an evader. Indeed, we show that with only the *minimal assumption of equal speeds*, the bounds for capture are much worse, for example $\Theta(\sqrt{n})$ vs. $\Theta(\log n)$ pursuers for simply connected polygons. However, by further assuming the maximum movement speed of the players is bounded by the features of the environment, upper bounds matching those of Guibas et al. [11] are obtained. Finally, we show that the minimal condition of equal speeds is enough for a randomized capture algorithm to match the bounds for randomized localization of Isler et al. [12].

2. Capture in simple polygons

In this section, we establish the tight bound of $\Theta(n^{1/2})$ for the number of pursuers needed to capture the evader when the environment is a simply-connected (hole-free) polygon of n vertices. We begin with some definitions and preliminaries that are commonly used throughout the paper.

We use e and p_i, respectively, to denote the evader and the ith pursuer. We will often use this notation to denote their positions as well. The players' sensing model is *visibility-based*: two players see each other only when they are in line of sight. The pursuit occurs in discrete-time but continuous-space: the players move in alternate turns (with the evader making the first move), and are allowed to move anywhere within the polygon subject only to the speed constraint. On their

[1] The results of these two papers have since appeared in a joint journal article [4].

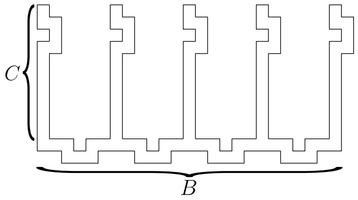

Fig. 1. Construction for the proof of Theorem 1.

turn, all the pursuers can move simultaneously, and independently. They can also freely communicate among themselves, to coordinate their moves and share their knowledge of the evader's position. All the players also know the geometry of the environment.

Without loss of generality, and with an appropriate scaling of the environment, we assume that the players' maximum speed is 1. Therefore, on its turn a player can move to any point that is within distance one of its current position, using the shortest path metric in the polygon. During the evader's turn, pursuers can see any portion of the evader's path that is within their line of sight. In the spirit of worst-case analysis, we assume that the evader has full knowledge of the pursuers' positions (even if they are not in its line of sight) and can even guess their future moves. All of our upper bounds hold even against such a strong evader.

We say that pursuers win the game if they can capture the evader in a finite number of moves irrespective of the latter's strategy. Conversely, the evader wins the game if it can elude the pursuers indefinitely. Our first theorem shows that, in the worst-case, at least $\Omega(n^{1/2})$ pursuers are needed to capture the evader in n-vertex polygons.

2.1. The lower bound construction

Theorem 1. *In an n-vertex simple polygon, at least $\Omega(n^{1/2})$ pursuers are needed in the worst-case to capture an equally fast evader.*

Proof. We give a construction of a polygon and the evader's strategy that forces $\Omega(n^{1/2})$ pursuers for a win. The polygon consists of a long corridor acting as a "base," of length B, with $n-1$ equally spaced "notches," and n long "channel corridors," interleaved with the notches (note the number of vertices is a constant multiple of n). See Fig. 1. Each channel corridor also has a notch at one end, and the length of each such corridor is C, chosen so that $C > B$. The players' maximum speed is set to $2C + B$, ensuring that players can move between the notches of two arbitrary channels in a single move, but pursuers cannot search more than two channels in one move: searching three or more channels requires speed of at least $4C$, which is strictly larger than $2C + B$. (The channel lengths take into account the notches, and we can scale the polygon as necessary to normalize the speed to unit speed.)

We now argue that capturing the evader in this polygon requires at least $\frac{1}{2}n^{1/2}$ pursuers. Given any placement of fewer than $\frac{1}{2}n^{1/2}$ pursuers, there exists a consecutive block of $n^{1/2}$ channel corridors and the section of base between them that does not have any pursuers. If the evader moves to a channel in this block, pursuers cannot determine the identity of the channel because the notches in the base block their visibility. Since each pursuer can search at most two corridors on its move, collectively these fewer than $\frac{1}{2}n^{1/2}$ pursuers cannot search all the $n^{1/2}$ corridors, leaving at least one safe corridor for the evader to hide. The evader can, therefore, continue to elude the pursuers indefinitely by repeatedly moving into such a "safe" corridor on its turn. This completes the proof. □

The rest of this section presents a matching upper bound, by giving an algorithm that guarantees a win for $O(n^{1/2})$ pursuers in all simply-connected polygons of n vertices. The next few subsections describe the geometric preliminaries and constructions that form the basis for our proof.

2.2. A k-block partition

We begin by describing a partition of the polygon P into subpolygons, called k-blocks, that plays an important role in our pursuit strategy. Each *k-block* is just a connected subpolygon of at most k vertices, and the partition satisfies the following properties:

1. the number of k-blocks in the partition is $O(n/k)$,
2. the edges common to adjacent k-blocks are polygon diagonals, and
3. the adjacency graph of the partition, called the *block graph*, is a binary tree.

Specifically, our k-block partition is an "unrefinement" of a triangulation of P: a triangulation partitions P into $(n-2)$ triangles, using $(n-3)$ diagonals; our partition retains $O(n/k)$ carefully chosen diagonals so that the resulting subdivision has the k-block partition properties. See Fig. 2 for an example. In fact, the degree bound of the adjacency graph is the only

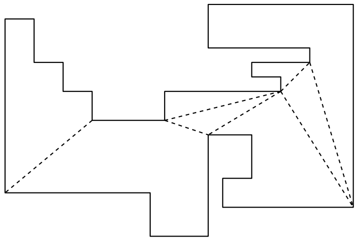

Fig. 2. A block partition of a polygon ($k = 8$).

non-trivial property—a naive partition can easily lead to unbounded fanout. We call the diagonals separating the k-blocks *cut edges*. The following lemma shows constructively that a k-block partition always exists.

Lemma 1. *Every simply-connected polygon on n vertices admits a k-block partition for any* $3 \le k \le n$.

Proof. Let T be a triangulation of P. The dual graph of the triangulation is a binary tree; the nodes of the graph are the triangles and its edges connect adjacent triangles. We describe a recursive algorithm to identify the *cut* edges that define the desired k-block partition. Since a subtree of size k corresponds to a subpolygon with $k+2$ vertices, we choose cut edges to break the tree into components of at most $k-2$ nodes, which form k-blocks.

We inductively assume that the tree is rooted at a degree one node r, which can initially be an arbitrary leaf node. For any node u in the tree, let $s(u)$ denote the size of the subtree rooted at u, including the node u itself. If $s(r) \le k-2$, then we are done. Otherwise, we choose any node u such that $s(r) - s(u) \le k-2$ but $s(r) - s(x) > k-2$ for any child x of u, and cut the edge between u and its parent. Next, if u has only one child x, then we simply recurse on the subtree T_u, but if u has both its children x and y, then we also cut the edges (u,x), (u,y) and then recurse on the subtrees $T_u \setminus (u,y)$ and $T_u \setminus (u,x)$, rooted at y and x, respectively. (During the recursive call, the size fields of the root nodes are recomputed for the new subtrees.)

For the correctness of the algorithm, we first note that the partition clearly creates valid k-blocks. Second, by construction, each block is bounded by at most three cut edges, thus ensuring that the block graph is a binary tree. Finally, to show that the total number of cuts made is $O(n/k)$, observe that in the block graph, any node of degree 1 or 2 is adjacent to a block so that the union of the neighboring blocks contains more than k vertices—otherwise, our algorithm will not have made the cut between the blocks. Since the number of degree 3 nodes is at most the number of leaves, the graph has size $O(n/k)$, and the proof is finished. □

After an initial search to localize the evader to a block, a placement of one pursuer per cut edge is sufficient to maintain the identity of the current block containing the evader. In particular, let $B(e)$ denote the current block containing the evader. Then the following lemma is straightforward.

Lemma 2. *Suppose a pursuer is placed on each cut edge of a k-block partition of the polygon. Then, after any move of the evader that crosses a block boundary, the pursuers know the identity of* $B(e)$*, the block containing the evader.*

The initial search can be performed using the following result of Guibas et al. [11]. We use the notation $\operatorname{diam}(P)$ for the *diameter* of the polygon under the shortest path metric.

Lemma 3. *(See [11].) Given a simply connected n vertex polygon* P*,* $O(\log n)$ *pursuers can locate the evader in* $O(n \cdot \operatorname{diam}(P))$ *moves.*

2.3. Critical moves

We begin with a simple sufficient condition to trigger the end-game: an immediate capture of the evader. Specifically, we say that an evader's move from its current position e to the new position e' is *critical with respect to a pursuer p* if there exists a point e_c on the evader's path that is *both* visible to p and closer to p's current position than to the evader's start position. Mathematically, a move from e to e' is critical for pursuer p at point e_c if (1) e_c lies on the path from e to e', (2) e_c is visible to p, and (3) $d(p, e_c) \le d(e, e_c)$. If the evader's move is critical with respect to k pursuers, we call it a *k-critical* move. Fig. 3(i) shows a k-critical event for $k = 4$, with f serving as the critical point. The following lemma shows the important connection between a k-block partition and a k-critical move.

Lemma 4. *Suppose each cut edge of the k-block partition is guarded by a pursuer, and a group of pursuers* $p_1, p_2, \ldots, p_k$ *are so positioned that an evader's move becomes k-critical with respect to these k pursuers. Then, one of the pursuers* p_i *can capture the evader on its next move.*

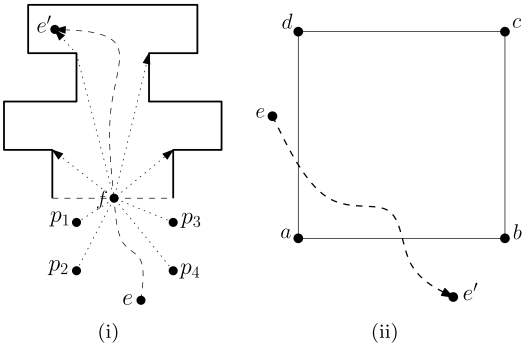

Fig. 3. In (i) illustration of proof of Lemma 4. In (ii) an example crossing sequence.

Proof. By definition of a k-critical move, for each pursuer p_i, there is a critical point, say, e_{c_i}, on the evader's path, closer to p_i than to the evader's start position. That is, $d(p_i, e_{c_i}) \le d(e, e_{c_i})$, for all i. (The critical points for different pursuers need not be the same.) By triangle inequality, we also have that

$$d\bigl(p_i, e'\bigr) \le d(p_i, e_{c_i}) + d\bigl(e_{c_i}, e'\bigr) \le d(e, e_{c_i}) + d\bigl(e_{c_i}, e'\bigr)$$
$$\le d\bigl(e, e'\bigr) \le 1$$

where the second inequality follows from the definition of a critical move and the last inequality from the unit maximum speed assumption. Thus, if the terminal position e' of the evader is *visible* from any of the critical points e_{c_i}, then the corresponding pursuer can capture the evader by first moving to its (visible) critical point and then to e'. Therefore, assume that none of the critical points are visible from the evader's position e'. In this case, we first move all the pursuers to a carefully chosen *waypoint* f, defined as follows. The waypoint f is the *last location at which the evader is seen by any pursuer* during its move from e to e'. After moving to the waypoint, each pursuer p_i still has $1 - d(p_i, f) \ge d(f, e')$ amount of remainder distance in its current move because $d(p_i, f) \le d(e, f)$. If the evader's terminal position e' is visible from f, the evader can be captured by any of the pursuers.

Thus, assume that the evader's location e' is invisible from the waypoint also. Suppose $B(e')$ is the block containing the evader, which the pursuers know by Lemma 2, and by construction $B(e')$ has at most k vertices. We first observe that the waypoint f must lie in $B(e')$: the evader's entry into $B(e')$ was seen by a cut edge pursuer. Consider the shortest path in $B(e')$ from f to the evader's final position e'—necessarily, this path cannot be longer than the portion of the evader's path between f and e', and because the two end positions are mutually invisible, the shortest path must contain at least one polygon vertex. Without loss of generality, let z be the *last* (closest to e') vertex on the path from f to e'. The vertex z is necessarily in $B(e')$, and is visible from e'. Since there are at most k choices for the vertex z, each of our k pursuers follows a shortest path from the waypoint f to one of these vertices (see Fig. 3(i) for illustration), and the one reaching z can successfully capture the evader. This completes the proof. □

2.4. Forcing a critical move

The main problem now is to devise a pursuer strategy that forces a k-critical move in a finite number of steps. Unfortunately, the cut edges can be arbitrarily longer than the normalized speed of the players, and thus even if we position k pursuers on an edge of length $L \gg k$, the evader can cross the edge without triggering a critical event. We, therefore, resort to a more complex structure and search strategy, which is motivated by the following simple observation: *If the evader crosses a square-shaped region of the environment with pursuers at its corners, then it is a critical move.*

In order to make this idea more precise, we first define a *crossing sequence*. Let R be a square contained entirely within the polygon P. A *crossing sequence* for R is a sequence of moves in which the evader enters and exits the square through *distinct edges of the square*. Fig. 3(ii) shows an example. (We note that an evader path is *not* a crossing sequence if it enters and exits the square through the same edge.)

Lemma 5. *Consider a square R fully contained in the polygon P, and let p_a and p_b, respectively, be two pursuers located at the corners a and b of R. Then, any crossing sequence in which the evader exits R through the edge (a, b) forces a critical event with respect to p_a or p_b.*

Proof. First, consider the simpler case when the crossing sequence consists of a single evader move: that is, the evader, originally outside the square, crosses it in a single move, exiting through the edge (a, b), say, at a point x. In this case, elementary geometry shows that $\min\{d(p_a, x), d(p_b, x)\} \le d(e, x)$, ensuring a critical event.

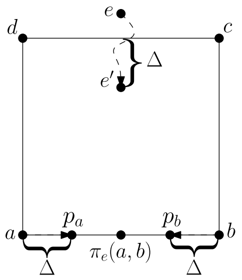

Fig. 4. Illustrating the proof of Lemma 5.

The case when the crossing sequence consists of multiple moves requires a more tedious, but still elementary, argument. See Fig. 4 for an illustration. We first introduce the idea of a projection. Given the evader's position e inside the square, its *projection on an edge* (a, b), denoted $\pi_e(a, b)$, is the point on (a, b) that is closest to e. (In other words, the projection is the foot of the perpendicular from e to (a, b).) The key observation is that a pursuer located at the projection $\pi_e(a, b)$ is *closer* to any point of (a, b) than the evader, and so any evader move crossing the edge (a, b) is critical for that pursuer. If the evader enters R through an edge adjacent to (a, b), namely, (a, d) or (b, c), then p_a or p_b can easily maintain their position on the evader's projection on (a, b): because the "horizontal projection" of the evader's position can change by at most one in a move, the pursuers p_a or p_b can reposition themselves at the evader's projection on the edge (a, b).

Thus, it remains only to consider the evader's entrance through the edge (c, d), which is the opposite side of the square from (a, b). In this case, clearly, both p_a and p_b can be arbitrarily far from the projection $\pi_e(a, b)$—the side length of R can be much larger than 1, the players' speed, and the evader may enter in the middle of the edge (c, d). However, in this case, the evader is also far away from the edge (a, b), and we ensure that a crossing sequence has a critical event for at least one of the pursuers. In order to track the pursuers' progress, let us introduce $\Delta = d(e, \pi_e(c, d))$, the distance between the evader's current position and its projection on the entrance edge (c, d).

The pursuer p_a now aims for the point that is $\min\{\Delta, d(a, \pi_e(a, b))\}$ away from a along the edge (a, b), while p_b aims for an analogous point measured from b. (See Fig. 4.) Each move of the evader can increase Δ by at most one, and so p_a and p_b can both reach their targets each turn. Once p_a or p_b reaches the projection, the claim of the lemma is clearly satisfied. Otherwise, both the pursuers are as close to the projection as is the evader, and together they guarantee that any move of the evader that causes it to exit through (a, b) is critical for both the pursuers. In particular, for any position of p_a in the interval between a and $\pi_e(a, b)$, the pursuer p_a guarantees a critical event, while p_b guarantees a critical event for any position in the interval from $\pi_e(a, b)$ to b. This completes the proof. □

An easy consequence of Lemma 5 is that by placing two pursuers at each vertex of R, for a total of eight, we can guarantee that any crossing sequence through the square R is critical move. (We could, in fact, reduce the number of pursuers to just four, one per corner but for convenience and a future simplicity, we choose to keep all eight pursuers.) One subtle point, however, is that the evader can exit through the same side it entered, therefore, *without completing a crossing sequence*, but in the process force pursuers to move off their desired corner positions! Our next lemma shows that the pursuers can immediately recover their initial positions, following any such "fake" move by the evader.

Lemma 6. *If the evader enters and exits the square R through the same side, without completing a crossing sequence, then all eight corner pursuers can recover their initial positions on the next move after the evader's exit.*

Proof. Without loss of generality, assume that the evader enters and exits R through the side (c, d), and that the exit move occurs at time t. By the projection invariant maintained by the pursuers, each of them is within distance 1 of its initial corner at time t, and therefore can recover its initial state on its next move. It is worth pointing out that if the evader exits through (c, d) but immediately reenters R through a different edge *in the same move*, then a crossing sequence is completed, immediately leading to capture—this follows from the projection invariant maintained by the pursuers. □

The idea of crossing sequence through a square extends easily to squares "truncated" by intersection with the polygonal environment. The intersection $R \cap P$ between the square R and the polygon P consists of possibly many (connected) cells. Consider one such cell F that does not contain any *vertex of* P, and call it *empty*. It is easy to see that F has a constant number of boundary edges—at worst, each of its corners can be lopped off by a polygon edge, resulting in an 8-sided cell. Thus, there are at most four sides of F inherited from R, each possibly truncated by a polygon edge, and at most four sides defined by polygon edges. Since the polygon edges are impenetrable by either the evader or the pursuers, it is easy to see that the critical move claim of the preceding lemma holds also for such a truncated cell F of the square R. In particular, we have the following easy corollary of Lemma 5.

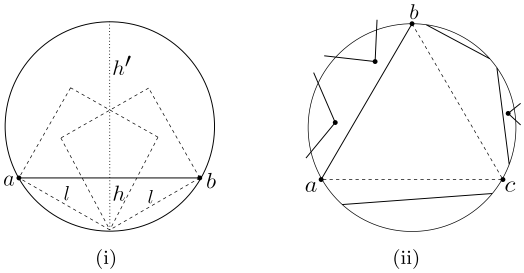

Fig. 5. (i) Illustrates the proof of Lemma 7 and (ii) shows an example triangle from a CDT.

Corollary 1. *Given a square $R = (a, b, c, d)$, let F be an* empty *cell of the common intersection $R \cap P$, let (a', b') be a non-polygon edge of F, and let $p_{a'}, p_{b'}$ be two pursuers placed at a' and b'. Then, any crossing sequence by the evader exiting F through the edge (a', b') is a critical move for one of these pursuers.*

2.5. Edge covers and the constrained Delaunay triangulation

We mentioned earlier that no bounded number of pursuers on a cut edge can prevent the evader from crossing it. Instead we build a geometric "cover" around each cut edge in such a way that the evader cannot cross the cover without being captured. We begin with the following technical lemma that forms the basis of such a cover.

Lemma 7. *Consider a circle C and a chord (a, b) in it. Then, there always exist two squares R_1, R_2 contained in C so that (a, b) lies in the union $R_1 \cup R_2$.*

Proof. Let r be the radius of the circle C. Let h and h' be the lengths of two segments into which (a, b) divides the circle's diametric chord that is perpendicular to (a, b). Without loss of generality, assume that $h \le h'$, which implies that $h \le r$. It is now easy to see that the two squares, each with side length $\ell = ((|ab|/2)^2 + |h|^2)^{1/2} \le r\sqrt{2}$, satisfy the conditions of our lemma, as illustrated in Fig. 5(i). Since every circle of radius r admits a contained square of side length $r\sqrt{2}$, these squares lie within C. □

Our idea is to cover each cut edge with the union of two squares as in Lemma 7, but use a particular kind of underlying triangulation to achieve the necessary empty-cell condition (cf. Corollary 1). Specifically, we use the *constrained Delaunay triangulation* of P [6,23] as the basis for our partition. The constrained Delaunay triangulation has the following properties: (1) each edge of the polygon appears as an edge of the triangulation, and (2) each triangle's circumcircle encloses no vertex that is visible from the interior of the triangle. Fig. 5(ii) shows an example.

Consider a cut edge (a, b) of the constrained Delaunay triangulation, which by definition has a circumcircle C empty of any visible vertices of P. By Lemma 7, we can find two squares R_1, R_2 that "cover" (a, b) and lie entirely within C. These squares may intersect the boundary of the polygon P but, by definition of CDT, the cells containing the edge (a, b) are empty of any visible vertices. We define the $cover(a, b)$ as the union of these two "truncated squares." These edge covers are utilized in the following way in our pursuit strategy.

Let B be a k-block partition, and B_i a k-block of B. Define $B_i^- = B_i \setminus \{\bigcup_j cover(a_j, b_j)\}$ as the *contracted block* corresponding to B_i, where (a_j, b_j) are the (at most three) cut edges bounding B_i. Similarly, we define $B_i^+ = B_i \bigcup_j \{cover(a_j, b_j)\}$ as the *extended block* corresponding to B_i. Since each edge cover has a constant number of vertices, all contracted or extended blocks clearly have size $O(k)$. We call two blocks (contracted or extended) neighbors if their original blocks share a common cut edge. We have the following lemma.

Lemma 8. *Any evader move between two neighboring contracted blocks is a crossing sequence.*

Proof. Let B_1^- and B_2^- be two contracted blocks neighboring the cut edge (a, b). Any sequence of moves by the evader from B_1^- to B_2^-, or vice versa, must cross at least one of the truncated squares of $cover(a, b)$. □

We say that a cut edge is *k-covered* if we replace each pursuer in $cover(a, b)$ by k co-located pursuers. Clearly, any evader move that crosses a k-covered edge is k-critical, and results in capture on pursuers' next move (cf. Lemma 4). At a high level our capture algorithm has the following form.

Algorithm. HoleFreeCapture

1. Construct a k-block partition of P, using the constrained Delaunay triangulation. Place one pursuer on each cut edge to track $B(e)$, the current k-block containing the evader.

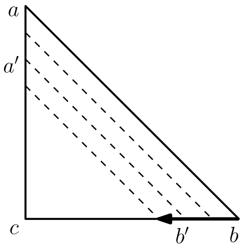

Fig. 6. The end game: shrinking the triangle during the attack phase.

2. Perform a sweep of the block graph until the evader is trapped in an extended block $B^+(e)$ whose adjacent cut edges are all k-covered. With the pursuers in this position, any move by the evader exiting $B^+(e)$ is a k-critical move, leading to capture.
3. With the evader confined to an extended k-block, we use an additional set of $O(k)$ pursuers to find and capture the evader in $B^+(e)$.

By choosing $k = n^{1/2}$, this leads to a search and capture strategy using $O(n^{1/2})$ pursuers: there are $O(n/k) = O(n^{1/2})$ cut edges, each requiring one pursuer, and at most 3 groups of $O(n^{1/2})$ pursuers needed to sweep the block graph. It only remains to describe the details of Steps 2–3, which is the focus of the next two lemmas.

Lemma 9. *With $O(k)$ pursuers, we can confine the evader to an extended block.*

Proof. We use the fact that the block graph is a binary tree, whose nodes correspond to the k-blocks and whose edges correspond to cut edges, and that pursuers always know $B(e)$, the block containing the evader, by Lemma 2. Inductively, assume that $B(e)$ belongs to the subtree rooted at a node u, and the cut edge between u and its parent is k-covered; if u is root, then the parent is null. Let x and y be the two children of u, one of them may be null. For notational convenience, we use U, X, Y to denote the blocks corresponding to u, x and y, respectively.

We begin by repositioning two groups of $O(k)$ pursuers each to form k-covers of the cut edges (u, x) and (u, y), and then use a constant number of pursuers to search the constant-size subpolygons *cover*(u, x) and *cover*(u, y). By the end-game algorithm (cf. Lemma 10), *if the evader is in these subpolygons*, it is either captured or forced to exit it. The key observation is that *once the evader has exited cover*(u, x) and *cover*(u, y), the pursuers in the cover prevent the evader from crossing between the neighboring contracted blocks. Thus, after leaving the covers if the evader remains in U^+, then it is confined: moving to a neighboring contracted block forces a k-critical event (cf. Lemma 8). Because all three neighboring cut edges of U^+ are k-covered, the claim follows in this case. Otherwise, the evader must have moved outside the extended block U^+. Without loss of generality, assume that the evader is either in the extended block X^+ or in a descendant node of x. In either case, the evader cannot enter U^- because of the k-covering of (u, x). We can therefore free up the k-covering pursuers from edges (u, v) and (u, y), and recursively search the subtree rooted at x. The search terminates within $O(n/k)$ such steps. □

2.6. The end game

The last step of our algorithm deals with capturing the evader when it is confined to an extended k-block, using $O(k)$ pursuers. At a superficial glance, it may appear that this can be done by combining the localization algorithm of Guibas et al. [11] with a modified lion-and-man algorithm of Isler et al. [12]. Unfortunately, this strategy fails due a technical subtlety: the lion-and-man algorithm of [12] relies on a *small step size* assumption, which precludes the arbitrary speed with which the evader is allowed to move in our problem. However, since we have $O(k)$ pursuers available to us, we can design a simple direct algorithm for this end game.

Lemma 10. *Let P be a k-vertex simple polygon. Then, in $O(\mathrm{diam}(P)^2)$ steps $O(k)$ pursuers can capture an equally fast evader.*

Proof. The algorithm operates in two phases: a *preparation* phase followed by an *attack* phase. The preparation phase begins by triangulating P. We then assign one pursuer to each edge of the triangulation, including the polygon boundary edges, which positions itself at the projection of the evader. We define the projection $\pi_e(a, b)$ of the evader for an edge (a, b) as the closest point on the edge to the evader *measured by direct Euclidean distance ignoring the polygonal boundary*—this is either the foot of the perpendicular from the evader's position or an endpoint a or b. Since the evader cannot leave P, and moves at most distance one in each move, each pursuer can arrive at its projection in $O(\mathrm{diam}(P))$ moves, which completes the preparation phase.

With each pursuer at the projection of its designated edge, the *attack* phase starts (see Fig. 6). Any move by the evader crossing a triangle is a critical move for some pursuer. Let p be a pursuer for whom this is a critical move. Since the entire polygon is collectively visible to the pursuers, the end position of the evader after the move is also known to p, and therefore p can capture the evader on its next move.

Thus, to avoid capture, the evader must remain confined to a single triangle, say, $\Delta(a, b, c)$. In this case, the three pursuers assigned to the triangle progressively "shrink" the area within which the evader lies, leading to eventual capture, as follows. Imagine sliding one of the edges of the triangle, say, (a, b) toward c by distance one, creating a "shrunken" triangle $\Delta(a', b', c)$. Position a new pursuer p' on the edge (a', b') at the projection $\pi_e(a', b')$ in $O(\text{diam}(P))$ moves. Now, if the evader lies in the strip between (a, b) and (a', b'), then a single pursuer can eventually capture the evader by sweeping this strip—the width of the strip is one, and the evader cannot cross the strip boundaries because they contain pursuers on projection points. If, however, the evader is in the triangle $\Delta(a', b', c)$, then we have successfully reduced the height of the triangle by one, which must lead to capture in $O(\text{diam}(P)^2)$ steps. This completes the proof. □

This completes our discussion of the algorithm HoleFreeCapture. As discussed earlier, by choosing $k = n^{1/2}$, we achieve the following theorem, which is the main result of this section.

Theorem 2. *$O(n^{1/2})$ pursuers are always sufficient to capture an equally fast evader in any simple polygon of n vertices in $O(n \cdot \text{diam}(P) + \sqrt{n} \cdot \text{diam}(P)^2)$ moves.*

Proof. As the capture will clearly occur, we need only analyze the worst-case number of moves. By invoking Lemma 3 the pursuers can determine the initial k-block containing the evader in $O(n \cdot \text{diam}(P))$ moves. Additionally, the pursuers will cover at most $O(\sqrt{n})$ cut edges taking at most $O(\sqrt{n} \cdot \text{diam}(P)^2)$ moves due to invocations of the end-game algorithm (Lemma 10) to force the evader out of k-covers. Finally, when the evader is captured in a k-block it takes $O(\text{diam}(P)^2)$ moves for a total of $O(n \cdot \text{diam}(P) + \sqrt{n} \cdot \text{diam}(P)^2)$ moves. □

3. Capture in polygons with holes

In this section, we extend our results to polygonal environments with holes, also called multiply-connected polygons. We assume that the polygon contains h disjoint polygonal holes, and the total number of vertices including the holes is n.

3.1. An $\Omega(n^{2/3})$ lower bound construction

We begin with a construction showing that in the worst-case at least $\Omega(n^{2/3})$ pursuers are needed to capture the evader. The proof follows the basic outline of Theorem 1, but requires a more complicated construction.

Theorem 3. *In the worst-case, at least $\Omega(n^{2/3})$ pursuers are needed to capture an equally fast evader in a multiply-connected polygon with n vertices.*

Proof. Our construction is based on a rectangular grid of r rows and c columns (see Fig. 7). We convert this into a polygon by making each edge of the grid into a narrow corridor, so that the resulting polygon has $r \cdot c$ (rectangular) holes. Place a small notch in the middle of each corridor to block visibility across the notch. Next, at the top boundary of the grid, place the "comb" construction of Theorem 1, uniformly spaced so that there are n/c channel corridors in each of the c columns of the grid (where n is a constant multiple of the number of vertices). We associate each group of n/c channel corridors with the grid column immediately preceding it. The height C of the comb corridors is chosen such that $C > WH$, where W and H, respectively, are the width and the height of the grid (taking into account the notch detours). Finally, the movement speed of the players is set to $2C + WH$, which is strictly smaller than $4C$. This speed allows the evader to move between any two channel corridors in a single move but no pursuer can search more than two such corridors.

Define an *extended column* as the subpolygon consisting of the chain of r notched corridors associated with a grid column together with its n/c channel columns. See Fig. 7. Given any placement of pursuers in the polygon, we call an extended column *uncovered* if no point of the extended column is visible to any pursuer. We claim that given any placement of $c/4$ pursuers in the polygon, (1) there are at least $3c/4$ uncovered extended columns, and (2) if $r = \sqrt{c}$, there is a group of $\sqrt{c}/2$ uncovered extended columns such that the evader can move between any two of them undetected.

The first claim follows from the fact that corridor notches limit a pursuer's visibility to at most one extended column. Since there are $c/4$ pursuers and c extended columns, at least $3c/4$ are uncovered. We prove the second claim by contradiction: assume the claim is false, and partition the uncovered extended columns into equivalence classes (groups) $g_1, g_2, \ldots, g_\ell$ such that the evader can move between two columns of the same group undetected but not between two columns of different groups. Because we have $3c/4$ uncovered extended columns, and by assumption each group g_i has fewer than $\sqrt{c}/2$ columns, the number of equivalence classes is $\ell > (3c/4)/(\sqrt{c}/2) = 3\sqrt{c}/2$. We can order these groups in their natural left-to-right order: all columns of one group must precede columns of the next group. We now claim that there must be at least one pursuer in every row between two consecutive groups: otherwise the evader can sneak between columns of two different groups, violating the equivalence class partition. Because the number of rows is $r = \sqrt{c}$, this implies there are at least $(3\sqrt{c}/2 - 1) \cdot \sqrt{c} = 3c/2 - \sqrt{c}$ pursuers, contradicting our initial assumption of at most $c/4$ pursuers. Thus, claims (1) and (2) are both true.

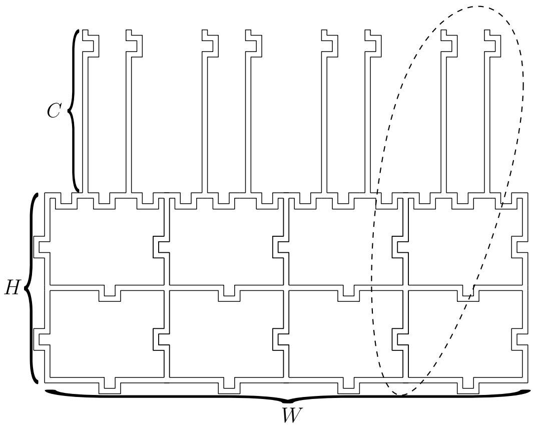

Fig. 7. The lower bound construction for capture in polygon with holes ($r = 2$ and $c = 4$). An extended column is shown with an ellipse around it.

Thus, there is a group of $\sqrt{c}/2$ uncovered extended columns for any placement of $c/4$ pursuers. The evader's strategy is to always move into one of these columns. Each of these columns contains n/c channel corridors for a total of $\frac{n}{2\sqrt{c}}$ channel corridors. If we choose $r = n^{1/3}$ and $c = n^{2/3}$, then $r = \sqrt{c}$, and we have $\frac{1}{2}n^{2/3}$ channel corridors for the evader to hide on its turn. Thus, if there are fewer than $c/4 = \frac{1}{4}n^{2/3}$ pursuers, the evader can indefinitely avoid capture by repeatedly moving into ones of the uncovered channel corridors that is not searched by pursuers on their turn. The entire polygonal environment, with holes, has $O(n)$ vertices, and this completes the proof that $\Omega(n^{2/3})$ pursuers are required in the worst-case. □

In the rest of the section, we present an upper bound for the number of pursuers needed to capture the evader in a polygon with holes.

3.2. A k-block partition of polygons with holes

We first extend the earlier notion of a k-block partition to polygons with holes. Our new partition has the structure of a planar graph, instead of a tree, and consists of two types of regions: separating triangles and k-block subpolygons (possibly with holes). The key property is that no two k-blocks are adjacent—they are adjacent only to separating triangles of the partition. More specifically, our partition satisfies the following properties:

1. the number of separating triangles (and blocks) in the partition is $O(n/k)$ if $h \le n/k$, and $O(\sqrt{nh/k})$ otherwise,
2. the adjacency graph of the partition is planar, and is called the *block graph*, and
3. every k-block of the partition has only separating triangles of the partition as its neighbors.

We construct such a partition through recursive calls to the well-known planar separator theorem.

Lemma 11. *(See [8].) Every planar graph $G = (V, E)$ on n nodes admits a partition of the nodes into three sets A, S, and B, such that neither A nor B has more than $2n/3$ nodes, S has at most $\sqrt{6n}$ nodes, and there are no edges with one endpoint in A and the other endpoint in B. The set of nodes S is called a* separator *of G.*

The only non-trivial part is that we want our separators to be of size $O(\sqrt{h})$, and not $O(\sqrt{n})$, but still want the two parts to be balanced in n. We do this by a suitable contraction of the triangulation graph of the polygon, and a recursive use of the separator theorem to achieve the balanced partition. We first need the following lemma as an intermediate result.

Lemma 12. *Given a triangulation of a polygon P with n vertices and h holes, we can find a set of $O(\sqrt{h})$ separating triangles whose removal partitions P into two (possibly disconnected) sub-polygons, each containing at most $2h/3$ holes and $2n/3$ vertices.*

Proof. The graph-theoretic *dual* of the triangulation is an $O(n)$ size planar graph, with a vertex for each triangle and an edge between two nodes if those triangles have a common boundary edge. In this graph, there is a *cycle* surrounding each of the h holes, and it is the structure of those cycles that is important to us. We reduce this triangulation graph to an $O(h)$ size planar graph, by repeatedly *contracting* vertices of degree 2, and deleting vertices of degree one, until all vertices have degree three. The resulting graph G has h faces, each vertex has degree 3, and so by Euler's formula, it has $h' = O(h)$ vertices and edges as well.

By the planar separation theorem, we can find a separator of size $O(\sqrt{h'})$ that splits the graph into two parts, each containing at most $2h'/3$ nodes, and by Euler's formula at most $2/3$ of the faces of G. In the primal space of triangulation, the separator corresponds to $O(\sqrt{h'})$ triangles, splitting the polygon into two pieces, each containing at most $2h'/3$ holes. However, the split does not guarantee any balance for the number of polygon *vertices*. Thus, if either piece contains more than $2n/3$ vertices, we apply the algorithm recursively until no piece has more than $2n/3$ vertices. The total number of triangles used to achieve the desired partition follows the recurrence $T(h') = T(2h'/3) + O(\sqrt{h'})$, with $T(1) = 1$, which solves to $T(h') = O(\sqrt{h'}) = O(\sqrt{h})$. In the base case, the subpolygon contains no holes, and a single triangle is sufficient to split the polygon into two pieces, each of size at most $2n/3$. This completes the proof. □

We repeatedly apply Lemma 12 to construct our k-block partition.

Lemma 13. *Every multiply-connected polygon with n vertices and h holes admits a k-block partition for any* $3 \le k \le n$.

Proof. We apply Lemma 12 recursively to our polygon P until each piece is a subpolygon of k vertices, possibly with holes. The recursive partition naturally corresponds to a binary tree, called the *partition tree*, whose leaves are the k-blocks and non-leaves are the separating triangles. The blocks corresponding to any two leaves necessarily are on opposite sides of a separator, and thus the partition has the desired adjacency property. It only remains to bound the total number of separating triangles and blocks. The number of blocks (leaves of the partition tree) is upper bounded by the number of non-leaf nodes plus one, which in turn is upper-bounded by the number of separating triangles. We now bound the number of separating triangles.

First, consider the case when the number of holes is $h \le n/k$. We classify the separating triangles into two groups, depending on whether or not the sub-polygon being split has holes. If the sub-polygon is hole-free, then a single separating triangle can split it in a balanced $\frac{1}{3}$–$\frac{2}{3}$ ratio, and so the total number of separating triangles used for splitting hole-free sub-polygons is $O(n/k)$. To bound the number of separating triangles used for splitting sub-polygons with holes, consider an intermediate sub-polygon P_j created during the partition, which has $h_j > 0$ holes. Call this sub-polygon *i-big* if $(2/3)^{i+1}h < h_j \le (2/3)^i h$. There are at most $(3/2)^{i+1}$ i-big subpolygons because the subproblems at any level of the partition tree are pairwise disjoint, and there are a total of h holes shared among them. The maximum value of i with an i-big sub-polygon is $\log_{3/2} h$, and since the separator of an i-big polygon has size $O(\sqrt{(2/3)^i h})$, a bound for the total size of the separators created during the partition tree is

$$\sum_{i=0}^{\log_{3/2} h} \left(\frac{3}{2}\right)^{i+1} \cdot c\sqrt{\left(\frac{2}{3}\right)^i h},$$

for some constant c. One can easily verify that this sums to $O(h)$. Thus, the total number of separating triangles in the partition is $O(h + n/k) = O(n/k)$.

Now, assume that $h > n/k$, and consider an intermediate sub-polygon P_j with n_j vertices and h_j holes. In this case, call the sub-polygon *i-big* if either $(2/3)^{i+1}h < h_j \le (2/3)^i h$ or $(2/3)^{i+1}n < n_j \le (2/3)^i n$. (Intuitively, a subpolygon is i-big if either its number of holes or its number of vertices is large enough to force a split to the next level.) We claim that there are at most $2 \cdot (3/2)^{i+1}$ subpolygons that are i-big. This holds because all subproblems at any level of the partition tree are pairwise disjoint, and at most $h/(2/3)^{i+1}h = (3/2)^{i+1}$ polygons arise due to the condition on the number of holes, and at most $n/(2/3)^{i+1}n = (3/2)^{i+1}$ due to the condition on the number of vertices.

Since $(2/3)^{\log_{3/2}(n/k)}n = k$, each non-leaf sub-polygon created during recursive partitioning is i-big for some i where $0 \le i < \log_{3/2}(n/k)$. Since the separator of an i-big polygon has size $O(\sqrt{(2/3)^i h})$, we can bound the total size of all the separators created during the partition tree as

$$\sum_{i=0}^{\log_{3/2}(n/k)} 2 \cdot \left(\frac{3}{2}\right)^{i+1} \cdot c\sqrt{\left(\frac{2}{3}\right)^i h},$$

for some constant c. One can easily verify that this sums to $O(\sqrt{nh/k})$, which completes the proof. □

The following lemma is straightforward.

Lemma 14. *Suppose a pursuer is placed in each separating triangle of a k-block partition of the polygon. Then, after any move of the evader that crosses a block or separating triangle boundary, the pursuers know the identity of* $B(e)$*, the block or separating triangle containing the evader.*

Additionally, the evader can be initially located using the following result of Guibas et al.

Lemma 15. *(See [11].) Given an n vertex polygon P with h holes,* $O(\sqrt{h} + \log n)$ *pursuers can locate the evader in* $O(n \cdot \mathrm{diam}(P))$ *moves.*

3.3. Analysis of capture in polygons with holes

We now have all the pieces in place to describe the outline of the capture strategy and derive our main result. Following our scheme for the polygons without holes, we construct the k-block partition using the *constrained Delaunay triangulation* of P so that any diagonal (edge of a separating triangle) can be covered using the construction of Lemma 7. In particular, we can k-cover all three edges of a separating triangle so that a k-critical evader move immediately leads to capture. At a high level, our capture algorithm has the following form.

Algorithm. PolygonWithHolesCapture

1. Construct a k-block partition of P, using the constrained Delaunay triangulation. Place one pursuer in each separating triangle to track the current block, or the separating triangle, $B(e)$ containing the evader.
2. Position pursuers at nodes of the partition tree until the evader is trapped in an extended block $B^+(e)$, whose adjacent separating triangles are all k-covered. With the pursuers in this position, any move by the evader exiting $B^+(e)$ is a k-critical move, leading to capture.
3. With the evader confined to an extended k-block, we use an additional set of $O(k + \sqrt{h})$ pursuers to find and capture the evader in $B^+(e)$.

Only Steps 2 and 3 require explanation—the k-block partition is already described by Lemma 13. Step 3, in fact, is also easy because the end-game algorithm of Lemma 10 works even with holes: a polygon with k vertices, including holes, can always be triangulated using $O(k)$ triangles, and our end-game algorithm requires a constant number of pursuers per triangle. We note that $B(e)^+$ has $O(k + \sqrt{h})$ vertices because in the worst-case, a block may neighbor $O(\sqrt{h})$ separating triangles. Thus, the only remaining part is Step 2, which is analyzed in the following lemma.

Lemma 16. *With $O(k\sqrt{h})$ pursuers, we can confine the evader to an extended block.*

Proof. We first position pursuers to achieve k-covering of each of the $O(\sqrt{h})$ separating triangles for the separator at the root node of the partition tree. We then search the $O(\sqrt{h})$ covers, which either leads to the capture or evicts the evader from these covers. The important observation is that once the evader is outside all the covers associated with the root's separator, it is confined to the (extended) blocks of one side of the separator—any crossing of the separator causes a k-critical event and leads to immediate capture.[2] Once the root node is covered, we recursively apply the algorithm to the child node whose subtree contains the block $B(e)$ with the evader, which the pursuers know by Lemma 14. The recursion stops when we reach a leaf node at which point the evader is confined to an extended k-block. Let us now examine the total number of pursuers needed in this search. Because the number of holes in a subproblem shrinks by factor 2/3 at each level, the number of pursuers needed to k-cover all the separators along a root-to-leaf path has the following recurrence: $T(h) = T(2h/3) + O(k\sqrt{h})$, which solves to $O(k\sqrt{h})$. □

We can now prove our main result.

Theorem 4. *Let $f(n, h)$ be the number of pursuers needed to capture an equally fast evader in a polygon of n vertices and h holes. Then,*

$$f(n,h) = \begin{cases} O(n^{1/2} \cdot h^{1/4}) & \text{if } h \le n^{2/3} \\ O(h^{1/2} \cdot n^{1/3}) & \text{otherwise} \end{cases}$$

Proof. By Lemma 16, we can confine the evader to a single extended k-block using $O(k\sqrt{h})$ pursuers, and then use the end-game algorithm to complete the capture with $O(k + \sqrt{h})$ additional pursuers. We choose the appropriate value of k, depending on the number of holes, to prove the result.

When $h \le n^{2/3}$, then we choose $k = n^{1/2}/h^{1/4}$. In this case, we have $h \le n/k$. The block partition has $O(n/k) = O(n^{1/2} \cdot h^{1/4})$ separating triangles, each requiring one pursuer. The k-covering of separating triangles requires $O(k\sqrt{h}) = O(n^{1/2} \cdot h^{1/4})$ pursuers. Thus, the total number of pursuers is $O(n^{1/2} \cdot h^{1/4})$.

When $h > n^{2/3}$, then we choose $k = n^{1/3}$. In this case, $h > n/k$, and the block partition has $O(\sqrt{nh/k}) = O(n^{1/3} \cdot h^{1/2})$ separating triangles, each requiring a single pursuer. The k-covering of separating triangles needs an additional $O(k\sqrt{h}) = O(n^{1/3} \cdot h^{1/2})$ pursuers, for the total of $O(n^{1/3} \cdot h^{1/2})$. This completes the theorem. □

The bounds of the preceding theorem can be combined into a single upper bound of $O(n^{5/6})$, giving the following theorem.

[2] Within a triangle. In that case, we treat the triangle as a trivial block.

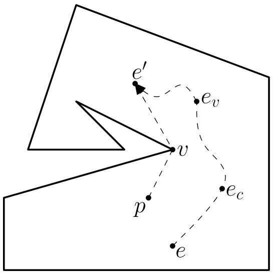

Fig. 8. The proof of Lemma 17.

Theorem 5. *Suppose P is an n vertex polygon with h holes, where n includes the vertices of the holes. Then $O(n^{5/6})$ pursuers can capture an equally fast evader in $O(n \cdot \mathrm{diam}(P) + \log(n) \cdot \mathrm{diam}(P)^2)$ moves.*

Proof. The worst-case number of pursuers is given as a corollary of Theorem 4, thus we concern ourselves with the duration of the capture. By invoking Lemma 15 the pursuers can determine the initial k-block containing the evader by in $O(n \cdot \mathrm{diam}(P))$ moves. Additionally, before confining the evader to a single k-block at most $O(\log n)$ sets of separating triangles must be k-covered, each taking $O(\mathrm{diam}(P)^2)$ moves when the evader must be evicted from a cover by Lemma 10. Thus, with the final invocation of the End Game algorithm to capture the evader in a k-block, the worst-case number of moves is $O(n \cdot \mathrm{diam}(P) + \log(n) \cdot \mathrm{diam}(P)^2)$. □

4. Minimum feature size assumption

We now show that a *minimum feature size* property of the environment is sufficient to yield significantly better upper bounds for the capture problem. Specifically, we show that $O(\log n)$ pursuers are always sufficient to catch the evader in a simply-connected polygon of n vertices, and $O(\sqrt{h} + \log n)$ if there are h holes. The pursuers' winning strategy is deterministic, and succeeds in polynomial time. The minimum feature size of a polygonal environment is defined as follows.

Definition 1 *(Minimum feature size (MFS)).* 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 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 (Fig. 1) violates the minimum feature size: the players' maximum speed is 1 but there are pairs of vertices that are within $1/\sqrt{n}$ of one another.

The primary reason that the minimum feature size allows a large reduction in the number of required pursuers is that a 1-critical move by the evader will result in its capture, *regardless of knowledge of which k-block contains the evader*. Meaning, a square can be guarded with eight pursuers, even if *no additional pursuers* are deployed in P. The following lemma proves this fact.

Lemma 17. *Suppose that the minimum feature size of P is at least one. Then if the evader's move becomes critical with respect to a pursuer p, then p can capture the evader on its next move.*

Proof. As shown in Lemma 4, if e's move to e' is critical for p at some point e_c, then it must be the case that $d(p, e') \le d(e, e') \le 1$. Thus, if the terminal position e' of the evader is *visible* to p, then p can capture the evader.

Let e_v be the last position during e's move where the evader is visible to p. Using the triangle inequality and the assumption $d(p, e_c) \le d(e, e_c)$, we conclude that $d(p, e_v) \le d(e, e_v)$. Notice then that the line segment (p, e_v) must contain a vertex of the environment, call it v, blocking p's visibility past the point e_v (see Fig. 8). We claim that the shortest path homotopic to (p, e_v, e') is (p, v, e'), that is, it consists of a single vertex v. Since the path (p, 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 $d(p, v) + d(v, e') \le 1$. The pursuer p, therefore, can capture by first moving to v and then to e' in a single move. □

Due to the preceding lemma, the pursuers need only 1-cover each cut edge, which can be done with $O(1)$ pursuers. Further, it is no longer necessary to position pursuers on the cut edges to track the current k-block of the evader as a 1-critical move is sufficient to capture the evader without this knowledge. As a result, when sweeping the block graph the pursuers will not know which subtree to recursively search after covering the cut edges (cf. Lemma 9). When this occurs, the pursuers use the localization strategy of Lemma 3 to find the evader, and determine which subtree to search. Thus, by setting $k = 3$ our algorithm will confine the evader to an extended 3-block, at which point $O(1)$ pursuers can capture the evader using the end game algorithm, and we obtain the following theorem.

Theorem 6. *Suppose P is a simply connected n vertex polygon with MFS at least 1. Then $O(\log n)$ pursuers can capture an equally fast evader in $O(n^2 \cdot \operatorname{diam}(P) + n \cdot \operatorname{diam}(P)^2)$ moves.*

Proof. Over the course of the algorithm at most three cut edges are covered at any one time, each using $O(1)$ pursuers. An additional $O(\log n)$ pursuers are reused to search for the evader, and finally $O(1)$ pursuers are used to capture the evader when it is confined to an extended 3-block. Thus $O(\log n)$ pursuers suffice to capture the evader.

For each recursive covering of cut edges in Lemma 9 one node is removed from the block graph and one search occurs. Thus, over the course of the algorithm, the evader must be located at most n times. By Lemma 3 the n searches each have duration $O(n \cdot \operatorname{diam}(P))$, and take a total of $O(n^2 \cdot \operatorname{diam}(P))$ moves. Additionally the end game algorithm may be invoked $O(n)$ times to force the evader out of edge covers and once for the final capture taking $O(n \cdot \operatorname{diam}(P)^2)$ moves. Thus the evader will be captured in $O(n^2 \cdot \operatorname{diam}(P) + n \cdot \operatorname{diam}(P)^2)$ moves. □

When P contains holes, we again take advantage of the fact a 1-critical move is sufficient for capture and cover separating triangles with $O(1)$ pursuers and set a block size of 3. Once again, due to the absence of pursuers on each cut edge, when sweeping the partition tree it is necessary to search each of the subtrees to determine which one the evader is in, which can be done with Lemma 15.

The following theorem bounds the total number of pursuers and moves needed to capture the evader.

Theorem 7. *Suppose P is an n vertex polygon with h holes and MFS at least 1. Then $O(\sqrt{h} + \log n)$ pursuers can capture an equally fast evader in $O(n^2 \cdot \operatorname{diam}(P) + n \cdot \operatorname{diam}(P)^2)$ moves.*

Proof. We make one slight modification to our algorithm used in Theorem 4; when the pursuit reaches a point where the evader has been confined to a simply connected polygon, stop covering separating triangles and apply Theorem 6. By Lemma 12 each separator splits the polygon into subpolygons with at most 2/3 as many holes, and thus the total number of separating triangles covered before reaching the simply connected polygon is $T(h) = T(2h/3) + O(\sqrt{h}) = O(\sqrt{h})$. Therefore $O(\sqrt{h})$ pursuers are needed to cover the separating triangles. Further, $O(\sqrt{h} + \log n)$ pursuers are reused to locate the evader, and $O(\log n)$ are used to capture the evader in a simply connected subpolygon for a total of $O(\sqrt{h} + \log n)$ pursuers.

The evader need only be located during each recursive partitioning, of which there are at most $O(\log(h))$, until a simply connected subpolygon is reached. The $O(\log(h))$ searches each have duration $O(n \cdot \operatorname{diam}(P))$ by Lemma 15 for a total of $O(n \cdot \log(h) \cdot \operatorname{diam}(P))$ moves. Additionally, the end game algorithm may be invoked $O(n)$ times to force the evader out of covered triangles taking $O(n \cdot \operatorname{diam}(P)^2)$ moves. Finally, invoking Theorem 6 takes $O(n^2 \cdot \operatorname{diam}(P) + n \cdot \operatorname{diam}(P)^2)$ moves. Thus the evader will be capture in $O(n^2 \cdot \operatorname{diam}(P) + n \cdot \operatorname{diam}(P)^2)$ moves. □

5. A randomized pursuit strategy

In our upper bounds so far, we have assumed that the evader can always predict the deterministic strategy of the pursuers. But now suppose that the pursuers have access to a source of randomness which the evader cannot predict. If they use this to randomize their movements, they can capture the evader in simply connected polygons with $O(1)$ pursuers, and $O(\sqrt{h})$ when there are h holes, even without the minimum feature size assumption.[3]

The first step to achieving these bounds is the following lemma that shows that a 1-critical move is sufficient for a pursuer to capture the evader with probability $1/n$.

Lemma 18. *If the evader's move is critical with respect to a pursuer p, then p can capture the evader on its next move with probability $1/n$.*

Proof. As e's move to e' is critical with respect to p, we know that $d(p, e') \leq 1$. Thus, if e' is visible to p, it is captured. Otherwise, there is at least one vertex on the shortest path from p to e'. Thus, suppose there are m vertices within distance one of p. The pursuer uniformly at random chooses and moves along the shortest path to one of those m vertices. With probability $1/m \geq 1/n$ the pursuer gains visibility of e and has moved along the shortest path to e. Thus p can use to remainder of its move to capture the evader. □

Suppose now that we use the MFS capture algorithm for simply connected polygons, except now edges are covered with probability $1/n$ using Lemma 18 (a crossing sequence results in capture with probability $1/n$). If the evader performed

[3] Applying the minimum feature size assumption reduces the expected time to capture by a factor of n, but does not change the number of required pursuers.

a critical move it would have $1/n$ chance of being captured (if no critical move is performed it will be captured). If the evader avoided capture after the critical move, the pursuers simply restart the algorithm. Thus, the evader is expected to be captured in n rounds of the algorithm.

In order to actually reduce the number of pursuers required to capture the evader, we replace the $O(\log n)$ pursuers with a single pursuer using the following randomized strategy of Isler et al. [12].

Lemma 19. *Given a simply connected n-gon, a single pursuer has a randomized strategy that can locate the evader in* $\operatorname{diam}(P)$ *moves with probability at least* $1/n$.

Using the preceding lemma, we are able to prove the following theorem.

Theorem 8. *Suppose P is a simply-connected n-vertex polygon. Then, $O(1)$ pursuers can capture the evader in $O(n^3 \cdot \ln(n) \cdot \operatorname{diam}(P) + n^2 \cdot \operatorname{diam}(P)^2)$ expected moves with probability at least $1 - \frac{1}{n}$.*

Proof. There are n expected rounds of the algorithm before the evader is captured. In each round, the evader must be located at most n times (by the same reason as Theorem 6), for a total of n^2 localizations. When locating the evader, the probability of success of a single trial is at least $1/n$. Using the inequality $(1+x) \le e^x$, one can easily show that after $3n\ln(n)$ trials of Lemma 19, the probability of not locating the evader is at most $1/n^3$. Then, by the union bound, the probability of failure in any of the n^2 localizations, is at most $n^2 \cdot 1/n^3 = 1/n$.

Thus, the n^2 localizations finish in $O(n^3 \cdot \ln(n) \cdot \operatorname{diam}(P))$ moves, with probability at least $1 - 1/n$. The remainder of the algorithm consists of executions of the end game algorithm and covering edges, both of which are deterministic, and their repetition over n rounds takes at most $O(n^2 \cdot \operatorname{diam}(P)^2)$ moves. Finally, $O(1)$ pursuers are used to cover the three cut edges, one pursuer is reused to locate the evader, and $O(1)$ to perform the end game algorithm and capture the evader, for a total of $O(1)$ pursuers. □

If P contains holes, we again use the MFS capture algorithm (Theorem 7) with the exception that separating triangles are covered with probability $1/n$. As in Theorem 7 the separating triangles are covered in order to confine the evader to a simply-connected polygon at which point the simply-connected algorithm is invoked (in this case the algorithm of Theorem 8). Finally, in order to reduce the number of required pursuers we replace the $O(\sqrt{h} + \log n)$ pursuer deterministic localization algorithm of Guibas et al. with the following randomized strategy of Isler et al. [12].

Lemma 20. *Given a multiply-connected n-gon with h holes, $O(\sqrt{h})$ pursuers have a randomized strategy which can locate the evader in $O(n \cdot \operatorname{diam}(P))$ moves with probability at least $1/n^2$.*

Using the preceding lemma, we are able to prove the following theorem.

Theorem 9. *Suppose P is a multiply-connected n vertex polygon with h holes. Then, $O(\sqrt{h})$ pursuers can capture the evader in $O(n^5 \cdot \ln(n) \cdot \operatorname{diam}(P) + n^2 \cdot \operatorname{diam}(P)^2)$ expected moves with probability at least $1 - \frac{1}{n}$.*

Proof. There are n expected rounds of the algorithm before the evader is captured. In the worst-case, the evader must be located n times per round, for a total of n^2 localizations.[4] During one localization, the probability of success of an $O(n \cdot \operatorname{diam}(P))$ move trial is at least $\frac{1}{n^2}$. Using the inequality $(1+x) \le e^x$, one can easily show that after $3n^2\ln(n)$ trials of Lemma 20, the probability of not locating the evader is at most $1/n^3$. Then, by the union bound, the probability of failure in any of the n^2 localizations, is at most $n^2 \cdot 1/n^3 = 1/n$.

Thus, the n^2 localizations finish in $O(n^5 \cdot \ln(n) \cdot \operatorname{diam}(P))$, with probability at least $1 - \frac{1}{n}$. The remainder of the algorithm consists of covering triangles, cut edges, and executing the end-game algorithm which are all deterministic, and their repetition over n rounds takes at most $O(n^2 \cdot \operatorname{diam}(P)^2)$ moves. Finally, $O(\sqrt{h})$ pursuers are used to guard the separating triangles, $O(\sqrt{h})$ are reused to locate the evader, and $O(1)$ pursuers by the algorithm of Theorem 8, for a total of $O(\sqrt{h})$ pursuers. □

6. Conclusion

In this paper, we studied a geometric pursuit-evasion problem in polygonal environments, and derived bounds on the number of visibility-guided pursuers needed to capture an equally capable adversarial evader. We proved a tight bound of $\Theta(n^{1/2})$ when the environment is a simply-connected polygon (no holes) with n vertices. When the environment has h holes, we showed a lower bound of $\Omega(n^{2/3})$, and two different upper bounds depending on the relative sizes of h and n. In

[4] Some of the localizations will be in the invocation of Theorem 8, however, considering them localizations among obstacles greatly simplifies the analysis at the cost of a slight increase in the time bound.

particular, for $h < n^{2/3}$, we gave an upper bound of $O(n^{1/2} \cdot h^{1/4})$, and for $h \geq n^{2/3}$ an upper bound of $O(n^{1/3}h^{1/2})$. Taken together, these can be simply expressed as a common upper bound of $O(n^{5/6})$.

Our results show that the visibility-based pursuit-evasion has a surprising and unexpected complexity. For instance, while $O(\log n)$ pursuers are always sufficient for the *evader detection* in simple polygons even if the evader is *arbitrarily faster* than the pursuers [11], $\Theta(n^{1/2})$ pursuers, each as fast as the evader, are needed for the evader capture. Similarly, while the bound for evader detection in polygons with holes is $O(\sqrt{h} + \log n)$, we show a lower bound of $\Omega(n^{2/3})$ for evader capture in the same setting. On the other hand, if the players' speed is small relative to the environment, formalized as a lower bound on the *minimum feature size condition*, then we have shown in that the capture bounds can be lowered to match the detection bounds. Further, even without this assumption, allowing the pursuers to make randomized moves the evader cannot predict can match Isler et al.'s [12] bounds for localization using a randomized algorithm.

Several open questions are naturally suggested by our work. The first and foremost is to close the gap between the upper and lower bounds for the polygons with holes. Additionally, finding a non-trivial lower bound for polygons subject to the minimum feature size constraint remains open.

References

[1] M. Adler, H. Räcke, N. Sivadasan, C. Sohler, B. Vöcking, Randomized pursuit-evasion in graphs, in: Proc. of the ICALP, SIAM, 2002, pp. 901–912.

[2] L. Alonso, Lion and man: upper and lower bounds, ORSA J. Comput. 4 (4) (1992) 447–457.

[3] D. Bhadauria, V. Isler, Capturing an evader in a polygonal environment with obstacles, in: Proc. of the IJCAI, 2011, pp. 2054–2059.

[4] D. Bhadauria, K. Klein, V. Isler, S. Suri, Capturing an evader in polygonal environments with obstacles: the full visibility case, Int. J. Robot. Res. 31 (10) (September 2012) 1176–1189.

[5] S.D. Bopardikar, F. Bullo, J. Hespanha, On discrete-time pursuit-evasion games with sensing limitations, IEEE Trans. Robot. 24 (6) (Dec. 2008) 1429–1439.

[6] L.P. Chew, Constrained Delaunay triangulations, in: Proc. of 3rd Symposium on Computational Geometry, ACM, New York, NY, USA, 1987, pp. 215–222.

[7] T.H. Chung, G.A. Hollinger, V. Isler, Search and pursuit-evasion in mobile robotics, Auton. Robots (2011) 1–18.

[8] H.N. Djidjev, On the problem of partitioning planar graphs, SIAM J. Algebr. Discrete Methods 3 (2) (1982) 229–240.

[9] A. Efrat, L.J. Guibas, S. Har-Peled, D.C. Lin, J.S.B. Mitchell, T.M. Murali, Sweeping simple polygons with a chain of guards, in: Proc. of the 11th ACM–SIAM Symposium on Discrete Algorithms, 2000, pp. 927–936.

[10] B.P. Gerkey, S. Thrun, G.J. Geoffrey, Visibility-based pursuit-evasion with limited field of view, Int. J. Robot. Res. 25 (4) (2006) 299–315.

[11] L.J. Guibas, J.-C. Latombe, S.M. LaValle, D. Lin, R. Motwani, Visibility-based pursuit-evasion in a polygonal environment, Int. J. Comput. Geom. Appl. 9 (5) (1999) 471–494.

[12] V. Isler, S. Kannan, S. Khanna, Randomized pursuit-evasion in a polygonal environment, IEEE Trans. Robot. 21 (5) (2005) 875–884.

[13] N. Karnad, V. Isler, Lion and man game in the presence of a circular obstacle, in: Proc. of the Int. Conference on Intelligent Robots and Systems, 2009, pp. 5045–5050.

[14] K. Klein, S. Suri, Complete information pursuit evasion in polygonal environments, in: Proc. of 25th Conference on Artificial Intelligence, 2011, pp. 1120–1125.

[15] K. Klein, S. Suri, Catch me if you can: pursuit and capture in polygonal environments with obstacles, in: Proc. of 26th Conference on Artificial Intelligence, 2012, pp. 2010–2016.

[16] K. Klein, S. Suri, Capture bounds for visibility-based pursuit evasion, in: Proc. of the 29th Symposium on Computational Geometry, SoCG '13, ACM, New York, NY, USA, 2013, pp. 329–338.

[17] S. Kopparty, C.V. Ravishankar, A framework for pursuit evasion games in R^n, Inf. Process. Lett. 96 (November 2005) 114–122.

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

[19] N. Megiddo, S.L. Hakimi, M.R. Garey, D.S. Johnson, C.H. Papadimitriou, The complexity of searching a graph, J. ACM 35 (1) (January 1988) 18–44.

[20] S.-M. Park, J.-H. Lee, K.-Y. Chwa, Visibility-based pursuit-evasion in a polygonal region by a searcher, in: Proc. of ICALP, Springer-Verlag, 2001, pp. 281–290.

[21] P.D. Seymour, R. Thomas, Graph searching and a min-max theorem for tree-width, J. Comb. Theory, Ser. B 58 (1) (May 1993) 22–33.

[22] J. Sgall, Solution of David Gale's lion and man problem, Theor. Comput. Sci. 259 (1–2) (2001) 663–670.

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

[24] I. Suzuki, M. Yamashita, Searching for a mobile intruder in a polygonal region, SIAM J. Comput. 21 (October 1992) 863–888.

[25] J. Thunberg, P. Ögren, A mixed integer linear programming approach to pursuit evasion problems with optional connectivity constraints, Auton. Robots 31 (4) (November 2011) 333–343.