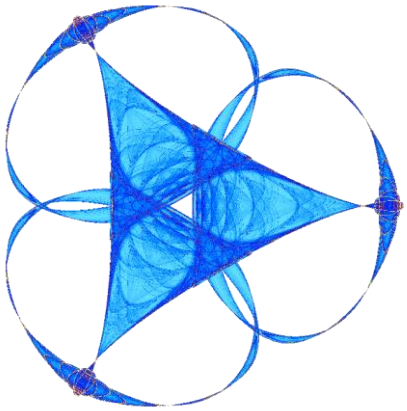
LINE-OF-SIGHT PURSUIT IN STRICTLY SWEEPABLE POLYGONS

By

Lindsay Berry, Andrew Beveridge, Jane Butterfield, Volkan Isler, Zachary Keller, Alana Shine and Junyi Wang

IMA Preprint Series #2456

(August 2015)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS
UNIVERSITY OF MINNESOTA
400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455-0436
Phone: 612-624-6066 Fax: 612-626-7370
URL: http://www.ima.umn.edu

Line-of-Sight Pursuit in Strictly Sweepable Polygons

Lindsay Berry* Andrew Beveridge† Jane Butterfield‡ Volkan Isler§
Zachary Keller¶ Alana Shine‖ Junyi Wang†

Abstract

We study a turn-based game in a simply connected polygonal environment Q between a pursuer $\mathcal{P}$ and an adversarial evader $\mathcal{E}$. Both players can move in a straight line to any point within unit distance during their turn. The pursuer $\mathcal{P}$ wins by capturing the evader, meaning that their distance satisfies $d(\mathcal{P}, \mathcal{E}) \leq 1$, while the evader wins by eluding capture forever. Both players have a map of the environment, but they have different sensing capabilities. The evader $\mathcal{E}$ always knows the location of $\mathcal{P}$. Meanwhile, $\mathcal{P}$ only has line-of-sight visibility: $\mathcal{P}$ observes the evader's position only when the line segment connecting them lies entirely within the polygon. Therefore $\mathcal{P}$ must search for $\mathcal{E}$ when the evader is hidden from view.

We provide a winning strategy for $\mathcal{P}$ in the family of *strictly sweepable polygons*, meaning that a straight line L can be moved continuously over Q so that (1) $L \cap Q$ is always convex and (2) every point on the boundary ∂Q is swept exactly once. This strict sweeping requires that L moves along Q via a sequence of translations and rotations. We develop our main result by first considering pursuit in the subfamilies of monotone polygons (where L moves by translation) and scallop polygons (where L moves by a single rotation). Our algorithm uses *rook strategy* during its active pursuit phase, rather than the well-known *lion strategy*. The rook strategy is crucial for obtaining a capture time that is linear in the area of Q. For monotone and scallop polygons, our algorithm has a capture time of $O(n(Q) + \text{area}(Q))$, where $n(Q)$ is the number of polygon vertices. The capture time bound for strictly sweepable polygons is $O(n(Q) \cdot \text{area}(Q))$.

1 Introduction

We study a turn based pursuit-evasion game in which a pursuer $\mathcal{P}$ chases an evader $\mathcal{E}$ in a simply connected polygon Q with boundary ∂Q. The players have unit speed, so they can move to any point within unit distance of their current location, provided that the line segment connecting these points is contained in Q. The pursuer's goal is to catch the evader in finite time using a deterministic strategy, while the evader tries to avoid capture indefinitely. Both players have full knowledge of the layout of environment, but they have asymmetric sensing capabilities. The evader has **full-visibility**, meaning

*Department of Mathematics, University of Texas, Austin TX 78712
†Department of Mathematics, Statistics, and Computer Science, Macalester College, St. Paul MN 55105
‡Department of Mathematics & Statistics, University of Victoria, Victoria, B.C. V8X 2X6
§Department of Computer Science & Engineering, University of Minnesota, Minneapolis MN 55455
¶Department of Mathematics, University of Minnesota, Minneapolis MN 55455
‖Department of Computer Science, Pomona College, Claremont CA 91711

that $\mathcal{E}$ knows the location of $\mathcal{P}$ at all times. The pursuer only has **line-of-sight visibility**: she knows the exact location of $\mathcal{E}$ only when the line segment $\overline{\mathcal{PE}} \subset Q$. Therefore, the pursuer must search when $\mathcal{E}$ is obscured by features of the environment. The capture condition requires both proximity and visibility: the evader is captured when $d(\mathcal{P}, \mathcal{E}) \leq 1$ and $\overline{\mathcal{PE}} \subset Q$.

The line-of-sight restriction puts the pursuer at a great disadvantage. Isler, Kannan and Khanna [8] provide a winning strategy for a full-visibility pursuer $\mathcal{P}$ in a simply connected polygon, called **lion's strategy**. They also construct environments for which a pursuer $\mathcal{P}$ with line-of-sight visibility does not have a deterministic winning strategy. Recently, Klein and Suri [10] provided an upper bound for the number of pursuers required for this setting: $O(n^{1/2})$ line-of-sight pursuers have a winning strategy in any simply connected polygon. They construct a comb-like environment (with visibility-blocking notches in its corridors) that requires $\Theta(n^{1/2})$ pursuers, so their bound is tight.

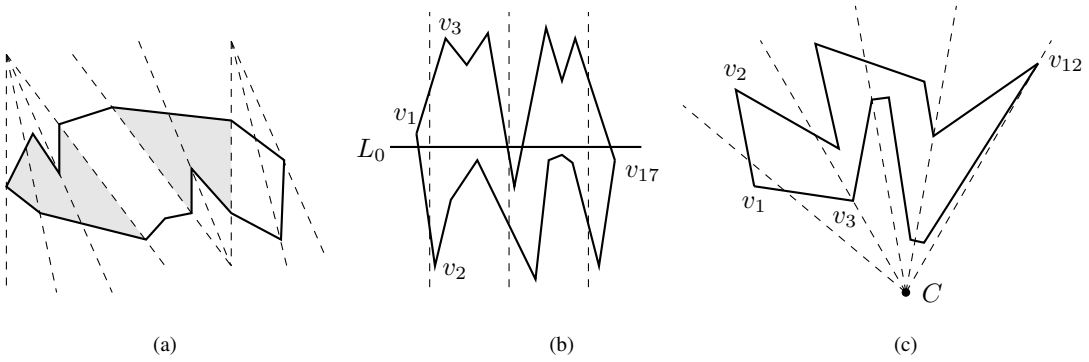


Figure 1.1: (a) A strictly sweepable polygon. (b) A monotone polygon with respect to horizontal axis L_0 with vertices indexed from left to right. (c) A scallop polygon with vertices indexed according to decreasing polar angle.

This paper addresses the other extreme: when can a single line-of-sight pursuer use a deterministic strategy to catch an evader? We provide a winning pursuer strategy for the family of strictly sweepable polygons. A polygon Q is **sweepable** if a straight line L can be moved continuously over Q so that $Q \cap L$ is always convex. The sweeping motion can include translations and rotations of the sweep line. A polygon is **strictly sweepable** if every point of Q is swept exactly once during this process, see Figure 1.1(a). We build up to this result, first describing winning pursuit strategies in two simpler subfamilies. A **monotone polygon** is a strictly sweepable polygon where the sweep line moves via translation along a fixed line L_0, see Figure 1.1(b) Our second family is **scallop polygons**, where the sweep line simply rotates through a fixed center C located outside of the polygon, see Figure 1.1(c). In this case, the total angle swept must be less than π to maintain the convex intersection with Q. Our main theorem summaries our results.

Theorem 1.1. *A line-of-sight pursuer can catch an evader in each of the following families of environments:*

(a) monotone polygons with capture time $O(n(Q) + \text{area}(Q))$,

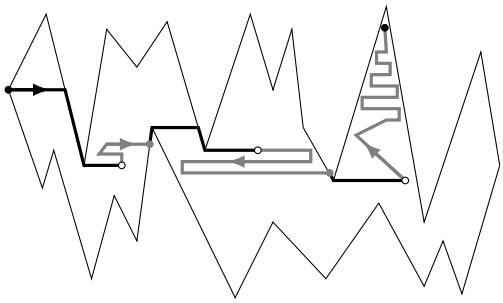


Figure 1.2: An example pursuit trajectory in which the evader makes two blocking moves. The white points indicate the transition from search mode to rook mode. The gray points indicate where the pursuer is blocked and transitions back to seach mode. The evader is captured at the final black point. Note that the pursuer never visits the same point twice.

(b) scallop polygons with capture time $O(n(Q) + \mathrm{area}(Q))$*, and*

(c) strictly sweepable polygons with capture time $O(n(Q) \cdot \mathrm{area}(Q))$.

Our pursuit algorithm alternates between a **search mode** and an active pursuit mode, called **rook mode**. During search mode, the pursuer methodically clears the polygon from left-to-right, maintaining a series of checkpoints that mark her progress. When $\mathcal{P}$ enters rook mode, her movements still guard these checkpoints. During rook mode, the evader can use features of the environment to hinder the pursuer. For example, the pursuer can be blocked by the boundary. In scallop and strictly sweepable polygons, the evader can also hide behind features. In response, $\mathcal{P}$ reverts back into search mode, carefully ensuring that she maintains her checkpoint progress. Maintaining the progress is straightforward in monotone polygons, but is trickier in scallop polygons and tricker still in strictly sweepable polygons. Eventually, the evader will be pushed into a subpolygon where the pursuer can capture him. Figure 1.2 shows a typical pursuit trajectory in a monotone polygon. The pursuer's searching phases are shown in black and the active pursuit phases are shown in gray. Note that searching always progresses from left-to-right. During active pursuit, $\mathcal{P}$ guards her current horizontal position, moving left or right to track the evader's movement, and advancing towards the evader when given the opportunity.

The proofs for each polygon family can be found in the three sections that follow, with our arguments building successively. Monotone pursuit is the simplest case, so we prove Theorem 1.1(a) in Section 2. Noori and Isler [16] proved that monotone polygons are pursuer-win for a line-of-sight pursuer. Their algorithm guarantees a capture time of $O(n(Q)^7 \cdot \mathrm{diam}(Q)^{13})$ where $\mathrm{diam}(Q) = \max_{x,y \in Q} d(x, y)$. Our pursuit strategy is much simpler, as is reflected in our significantly reduced capture time. The key improvement is a novel chasing strategy for pursuit in polygonal environments. The standard endgame in a polygon uses the **extended lion's strategy** [8] to capture the evader. Herein, we use a different tactic that is better suited to the monotone setting. This **rook's strategy** chases the evader while simultaneously guarding a horizontal frontier from incursion by the evader. This allows for a seamless transition between searching for the evader and actively chasing him. More specifically, our active rook phase also guards the pursuer's progress, while Noori and Isler needed a

third mode to defend against certain pursuer gambits, which increased their capture time bound.

In Section 3, we adapt the monotone pursuit strategy for scallop polygons. The guiding principle of the scallop strategy is to update the pursuer's frame of reference as she moves through the polygon. This update takes some care, as $\mathcal{P}$ must reposition herself to continue to guard her progress. Finally, we generalize to strictly sweepable polygons in Section 4, where our methodical analysis of monotone and scallop polygons pays off. We treat the sweepable environment as a progression of monotone and scallop polygons. The delicate transitions between these types requires even more care, since they can interact in subtle ways. Before launching into the proofs, we discuss related work and some environmental assumptions.

1.1 Related work and environmental assumptions

The literature on pursuit-evasion games spans a variety of settings and disciplines, with various motion and sensing capabilities. For a given environment, the main questions are (1) to determine the minimum number of pursuers needed to capture the evader, and (2) to bound the **capture time**, which is the number of rounds needed to catch the evader. Diverse game models have been studied, considering different types of environments, motion constraints, sensing capabilities and capture conditions. We list a sampling of these variations. Researchers have explored speed differentials between the players, constraints on acceleration, and energy budgets. As for sensing models, the players may have full information about the positions of other players, or they may have incomplete or imperfect information. Typically, the capture conditions requires achieving colocation, a proximity threshold, or sensory visibility (such as a non-obstructed view of the evader).

The original pursuit-evasion game seems to be Rado's lion-and-man game from the 1930's. In this game, lion chases man in a circular arena, with each player moving with unit speed. Lion wins if it becomes colocated with man in finite time, while the man tries to evade the lion forever. Intuition suggests that lion should be victorious. However, Besicovich showed that if the game is played in continuous time, then man can gently spiral away from the center, so that the lion becomes arbitrarily close, but never achieves colocation [12]. Meanwhile, the turn-based version time avoids this pathology: Sgall showed that the lion is victorious in finite time [19].

Pursuit-evasion games have enjoyed significant attention in the last decade, from three camps. The computational geometry viewpoint has been embraced by robotics researchers and theoretical computer scientists. In this setting, the turn-based pursuit game is played in a polygonal environment, a productive setting for studying autonomous agents. The game takes place inside a polygon with a finite collection of polygonal holes; the papers [7, 11, 13] provide introductions to pursuit in this setting. Isler et al. [8] introduced the **extended lion's strategy**, which adapts the original lion's strategy in circular environments. This is a winning strategy for a single pursuer in any any simply connected polygon. More recently, Noori and Isler [14, 15] employed a novel pursuer strategy called **rook's strategy** for pursuit on surfaces and convex terrains. This strategy, which was originally developed for the full visibility case, is valid whenever the capture distance is nonzero. Herein, we extend rook's strategy for line-of-sight pursuit in sweepable polygons. Interest in polygonal pursuit games has been spurred by the widespread availability of practical sensing technologies and robotics platforms. Applications for

automated search and pursuit are on the rise, including intruder neutralization, search-and-rescue, and tracking of tagged wildlife. Analysis of games with an adversarial evader provide important worst-case bounds for these applications.

Meanwhile, combinatorics researchers have devoted intense scrutiny to pursuit-evasion on graphs, where it is known as the game of cops and robbers. This graph game was introduced in the 1980's [18, 17]. For an overview of pursuit-evasion on graphs, see the monograph [6]. More recently, topologists have gotten into the game, characterizing capture strategies for spaces with various curvature conditions [2, 3]. There has certainly been cross-pollination between these traditions. For example, the analog of the classic Aigner and Fromme result that three cops can always capture a robber on a planar graph [1] was recently proven to hold for the polygonal setting by Bhaudauria et al. [5], and further generalized to a class of compact two-dimensional domains by Beveridge and Cai [4].

We conclude this section with comments on two aspects of our game model. First, we assume that there is a relationship between the geometry of the environment and the speed of the players. In particular, the environment has a **minimum feature size**, meaning that the minimum distance between any two polygonal vertices is at least one unit. This assumption can be achieved by a simple rescaling of the environment, if necessary. Klein and Suri [9] pointed out that this minimum feature size has been an implicit assumption in many papers. Furthermore, it is necessary to avoid an unexpectedly powerful evader who can use his super-speed to confound a line-of-sight pursuer. Of course, this assumption is a natural one: we view the turn-based game as an approximation of the continuous time, so we should choose a time scaling that appropriately partitions the dynamics of movement in the environment.

Second, recall that capture in our game requires two conditions: $d(\mathcal{P}, \mathcal{E}) \leq 1$ and $\mathcal{P}$ sees $\mathcal{E}$. We employ unit capture distance to simplify the presentation of our proofs. Our results also hold when we replace the first condition with $d(\mathcal{P}, \mathcal{E}) \leq \epsilon$ for any positive capture radius $\epsilon > 0$. The only impact of a smaller capture radius ϵ is a multiplicative factor of $1/\epsilon$ in the capture time. We note that the polygonal literature uses proximity capture ($d(\mathcal{P}, \mathcal{E}) \leq \epsilon$) and colocation capture ($d(\mathcal{P}, \mathcal{E}) = 0$) with roughly equal frequency. Herein, we take additional advantage of the former capture condition in developing the rook's strategy for chasing the evader. In particular, the buffer distance between the players plays an essential role in the rook's strategy. In practice, the players would have non-zero radius so this assumption always holds.

Finally, we assume that $|\mathrm{area}(Q)| \geq |\mathrm{diam}(Q)|$ throughout to simplify our time bound notation. This will generally be the case, except for long, skinny environments. Note that $\mathrm{area}(Q) \leq \mathrm{diam}(Q)^2$ whenever $\mathrm{diam}(Q) \geq 1$, so we could use the latter value for our capture time bounds, if desired. We state our bounds in terms of area since our analysis will naturally partition the polygon into disjoint regions.

2 Pursuit in a Monotone Polygon

In this section, we prove Theorem 1.1(a): $\mathcal{P}$ can capture $\mathcal{E}$ in a monotone polygon. We start by setting some notation. We fix our monotone axis L_0 and then label our vertices from left to right as $v_1, v_2, \ldots, v_n$. The monotonicity of Q has a simple characterization with respect to ∂Q: the boundary

can be partitioned into two piecewise linear functions defined on $[x(v_1), x(v_n)]$. These functions only intersect at v_1 and v_n, so we refer to them as the *upper chain* Π_U and the *lower chain* Π_L.

Given a point Z, let $x(Z)$ and $y(Z)$ denote its x- and y-coordinates respectively, so that $Z = (x(Z), y(Z))$, where the horizontal x-axis aligns with the monotone axis L_0. The horizontal and vertical line segments through Z that intersect ∂Q are denoted by $X(Z)$ and $Y(Z)$, respectively. We opt for this notation since we will view these lines as the "x-axis" and "y-axis" for the frame of reference centered at Z. The symbols $\mathcal{P}$ and $\mathcal{E}$ denote the pursuer and the evader respectively. We are often concerned with these positions at some time $t \geq 0$, and will use $\mathcal{P}_t = (x(\mathcal{P}_t), y(\mathcal{P}_t))$ and $\mathcal{E}_t = (x(\mathcal{E}_t), y(\mathcal{E}_t))$ to denote these points in the polygon. We define $\Delta x(t) = |x(\mathcal{P}_t) - x(\mathcal{E}_t)|$ and $\Delta y(t) = |y(\mathcal{P}_t) - y(\mathcal{E}_t)|$. Occasionally, we will drop the subscript and use $\mathcal{P}, \mathcal{E}$ to denote the player's positions, in order to ease the exposition.

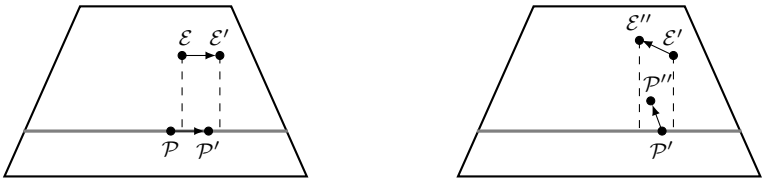
The high-level pursuer strategy is given in Algorithm 1. We start with a qualitative description of the pursuit, tackling the details of this algorithm in the subsections that follow. Our pursuer algorithm in a monotone polygon has much in common with the algorithm presented by Noori and Isler [16]. Both alternate between a searching mode and a chasing mode. In the search mode, $\mathcal{P}$ traverses left-to-right along a specified search path. At certain times, $\mathcal{P}$ is able to mark a region to her left as **cleared**, meaning $\mathcal{E}$ cannot enter this region without being captured. These checkpoints are necessary for showing that the pursuit terminates in finite time. This methodical advancement continues until $\mathcal{P}$ establishes the criteria to enter chase mode. This requires (1) visibility of $\mathcal{E}$ and (2) being positioned in a manner that protects the cleared region. Once $\mathcal{P}$ has transitioned into chase mode, she pursues the evader until $\mathcal{E}$ makes a rightward move that obstructs her pursuit (either by hiding behind a vertex or by using the boundary to block the pursuer's responding move). This forces $\mathcal{P}$ to re-enter search mode. Crucially, this is done in a way that protects (and perhaps =updates) the cleared territory. The algorithm continues, and the evader territory shrinks over time. Eventually, $\mathcal{E}$ is trapped in a subregion where he cannot foil the chase mode any longer, and $\mathcal{P}$ captures him.

Algorithm 1 Pursuit Strategy

Require: $\mathcal{P}$ starts out at left endpoint v_1

```
1: while E has not been captured do
2:     Create Search Path from the current location of P
3:     while P has not attained rook position do
4:         Follow Search Strategy
5:     end while
6:     while E does not make an escape move do
7:         Follow Rook Strategy
8:     end while
9: end while
```

The fundamental difference between our algorithm and that of Noori and Isler is the choice of chase mode. The method of chasing informs the trajectory for searching, so the search paths are also different. Noori and Isler use the extended lion's strategy to chase the evader. In lion's strategy, the pursuer stays on the shortest path between the leftmost point v_1 and the evader. Noori and Isler need



(a) (b)

Figure 2.3: Rook strategy in a convex polygon. (a) $\mathcal{P}$ is in rook position with negative offset $x(\mathcal{P}) - x(\mathcal{E}) \leq 1/2$. $\mathcal{P}$ can maintain this offset when $\mathcal{E}$ moves to the right. (b) Eventually, $\mathcal{P}$ will make progress, perhaps by switching to a positive offset when $\mathcal{E}$ finally moves to the left.

an additional guarding mode to establish this position after spying the evader. This guarding mode is quite involved (and contributes to their worst case $O(n(Q)^7 \cdot \mathrm{diam}(Q)^{13})$ capture time), as the evader has multiple gambits to threaten the pursuer's cleared region. However, once $\mathcal{P}$ has established lion's position, she can reliably push the evader to the right, ending in capture or an evader hiding move (which triggers a transition back to search mode).

Herein, we use a different chasing tactic called **rook's strategy** that is better suited for our monotone setting. This strategy has been used successfully on polyhedral surfaces for the full visibility case [14, 15], but this marks its first appearance for polygonal pursuit and with visibility constraints. Rook's strategy draws inspiration from the chess endgame of a black rook and black king versus a solitary white king. During this endgame, the black rook reduces the area available to the white king, one row at a time (with occasional support of the black king). Eventually, the white king is confined to a single row, where black can checkmate. At any given time, the rook guards a horizontal row on the chessboard. Likewise, a pursuer in rook position will also guard a horizontal frontier. This is a natural complement to the pursuer's horizontal search trajectory. Rook's strategy eliminates the challenging transition that Noori and Isler mitigated with their guarding phase, which dealt with the mismatch of their linear search path and the curved boundary of their cleared region. Our improved capture time of $O(n(Q) + \mathrm{area}(Q))$ directly reflects the seamless transition between our search mode and our rook mode.

We say that $\mathcal{P}$ is in **rook position** when (a) Q contains the line segment joining $\mathcal{P}$ and $\mathcal{E}$, and (b) $|x(\mathcal{P}) - x(\mathcal{E})| \leq 1/2$, see Figure 2.3(a). We refer to $x(\mathcal{P}) - x(\mathcal{E})$ as the pursuer **offset** and we refer to the horizontal line segment $X(\mathcal{P})$ as the **rook frontier**. In a convex polygon (where visibility is not an issue), rook position has three important characteristics. First, a pursuer in rook position can re-establish this position after every evader move. Second, the evader cannot get too close to the rook frontier (or cross it) without being captured. Third, the pursuer can reliably advance her rook frontier: the worst case scenario is when $\mathcal{E}$ moves full speed horizontally across the region. At some point, he must reverse direction, and the pursuer changes her offset (say, from positive to negative), and moves her rook frontier towards the evader, see Figure 2.3(b). The situation is more complicated in a general polygon, but these three basic characteristics remain the driving force of rook's strategy.

Figure 2.4 shows the eight different configurations for a pursuer in rook position. We say that $\mathcal{P}$ is

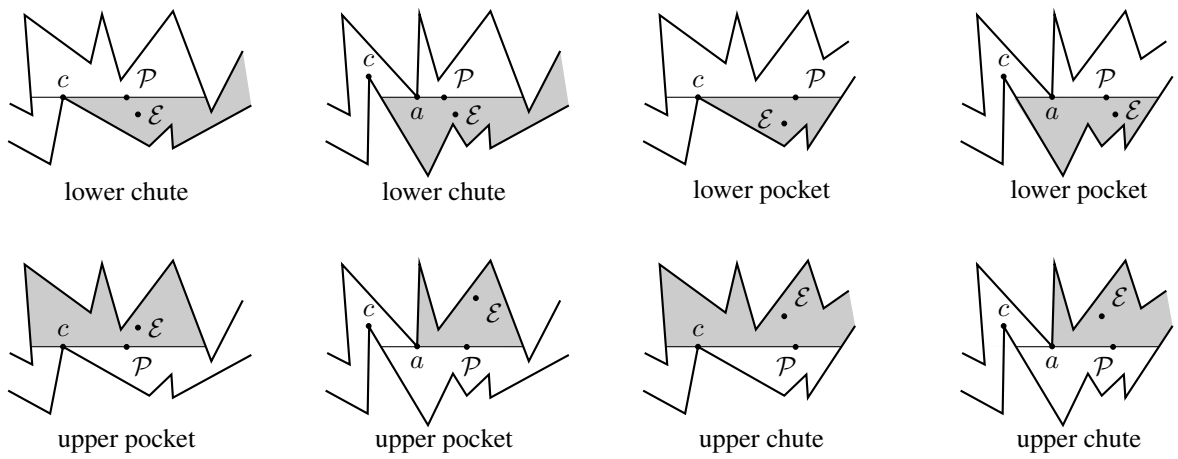


Figure 2.4: The eight types of rook positions. The evader territories are shown in gray.

in **lower (upper) pocket position** when (1) the evader is below (above) the pursuer, and (2) both of the endpoints of the rook frontier are on the lower chain Π_L (upper chain Π_U). We say that $\mathcal{P}$ is in **lower (upper) chute position** when (1) the evader is below (above) the pursuer, and (2) the left endpoint of the rook frontier is on the lower (upper) chain and the right endpoint is on the upper (lower) chain.

When the pursuer establishes pocket position, we have reached the endgame of the pursuit. The evader can try to use the features in the pocket to his advantage, but the pursuer always has a counter move. Eventually, the evader will be caught, just as in a convex polygon. On the other hand, when the pursuer establishes chute position, the evader can employ an **escape move** to prolong the game. For example, in a lower chute, $\mathcal{E}$ can move rightward and use the upper boundary Π_U to block the pursuer's rook move. In response, $\mathcal{P}$ transitions back into search mode, using a new search path drawn from her current location, see Figure 2.9.

This concludes the overview of our pursuer strategy. We work through the details in the subsections that follow.

2.1 Search Strategy

The key to the pursuer's search mode is the choice of **search path** Π. Algorithm 2 explains how to construct Π from any point $p \in Q$. For our original search path, $p = v_1$. Intuitively, the pursuer follows a search path Π that remains as horizontal as possible, the search path always terminates at the rightmost vertex v_n. An example of the initial search path is shown in Figure 2.5 (a).

Algorithm 3 describes the pursuer search movement. Typically, $\mathcal{P}$ traverses the search path at unit speed. However, she stops at the x-coordinate of each vertex of Q in order to (1) avoid stepping past an evader hiding behind a feature and (2) allow for a change in direction when encountering a vertex of ∂Q that is on the search path Π (recall that the pursuer can only move in a straight line on each turn). The pursuer also ensures that $x(\mathcal{P}) \leq x(\mathcal{E})$ at all times.

The pursuer keeps track of her **search frontier** $X_t = X(\mathcal{P}_t)$ and her **checkpoint** c_t, which is the

Algorithm 2 Create Monotone Search Path

Input: $p \in Q$
while $p \neq v_n$ **do**
    Move p horizontally until reaching the boundary ∂Q
    if p is on lower boundary Π_L **then**
        Traverse upwards along Π_L until reaching local maximum vertex $v_k \in \Pi_L$
    else
        Traverse downward along Π_U until reaching local minimum vertex $v_k \in \Pi_U$
    end if
end while

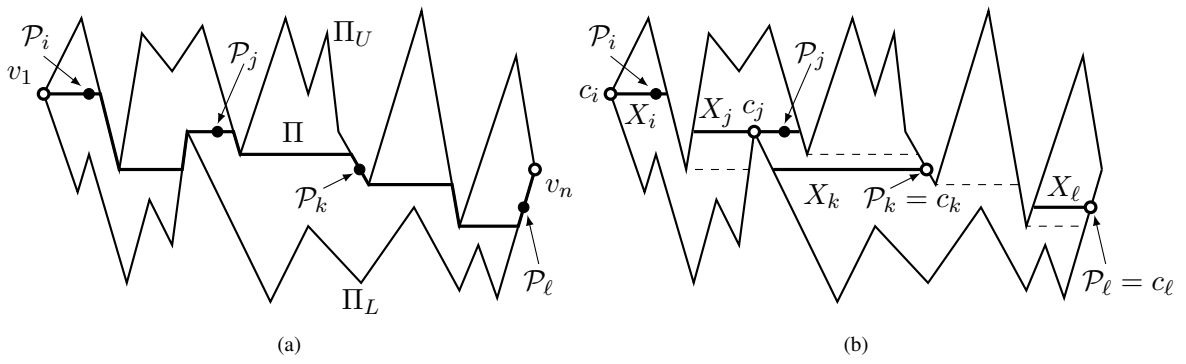


Figure 2.5: (a) Search path through a monotone polygon with four pursuer positions shown, $i < j < k < \ell$. (b) The corresponding search frontier X_t and checkpoint c_t for those four pursuer positions.

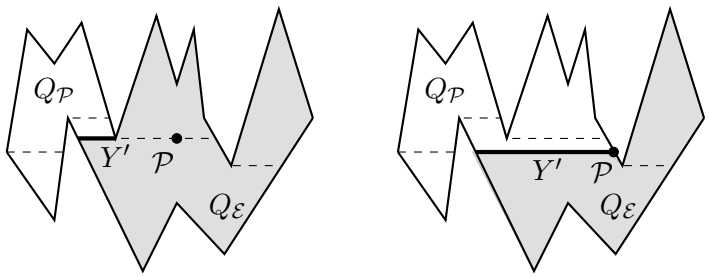


Figure 2.6: Two examples of a guarded frontier. The evader territory $Q_{\mathcal{E}}$ is shaded and pursuer territory $Q_{\mathcal{P}}$ is unshaded; the guarded frontier Y' is in bold.

point in $X_t \cap \partial Q$ that is closest to $\mathcal{P}_t$ among points to the left of $\mathcal{P}_t$; see Figure 2.5(b). Let $X'_t \subset X_t$ denote left part of the search frontier, up to and including c_t. We prove below that $\mathcal{E}$ cannot cross X'_t without being captured, so we call X'_t the **guarded frontier**. The guarded frontier partitions Q into two sub-polygons: the **pursuer territory** $Q_{\mathcal{P}}(t)$ to the left, and the **evader territory** $Q_{\mathcal{E}}(t)$ to the right; see Figure 2.6.

The checkpoint represents our progress. When the checkpoint location is updated, its x-coordinate increases, causing the pursuer territory to increase and the evader territory to decreases. In Section 2.3, we discuss this notion of progress more rigorously. Notice that $c_t = v_1$ until $\mathcal{P}$ first reaches ∂Q. Up until that event, the pursuer territory is simply the point v_1. Finally, we observe that checkpoint c_t is the unique rightmost vertex in the pursuer territory $Q_{\mathcal{P}}(t)$. Similarly, the left endpoint of the guarded frontier X'_t is the unique leftmost point in the evader territory $Q_{\mathcal{E}}(t)$. Both of these facts play a role in the pursuer's ability to guard $Q_{\mathcal{P}}(t)$.

The next lemma shows that if $\mathcal{E}$ moves horizontally past $\mathcal{P}$, then he must be visible.

Lemma 2.1. *Suppose that $\mathcal{P}$ is following the search strategy and $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_{t-1})$. If $x(\mathcal{E}_t) < x(\mathcal{P}_{t-1})$ then $\mathcal{P}_{t-1}$ can see $\mathcal{E}_t$. Furthermore, if $\mathcal{E}$ also crossed the horizontal line $X(\mathcal{P}_{t-1})$, then he is captured immediately.*

Proof. First, suppose that both $\mathcal{E}_{t-1}, \mathcal{E}_t$ are invisible to $\mathcal{P}_{t-1}$. We have $x(\mathcal{E}_t) < x(\mathcal{P}_{t-1}) < x(\mathcal{E}_t)$, so each evader position is obscured by a distinct vertex. However, these two vertices would be separated by a distance strictly less than $d(\mathcal{E}_t, \mathcal{E}_t) \leq 1$, which is impossible due to the feature size of Q. Therefore, at least one of $\mathcal{E}_{t-1}, \mathcal{E}_t$ is visible to $\mathcal{P}_{t-1}$. Without loss of generality, assume that $y(\mathcal{P}_{t-1}) \leq y(\mathcal{E}_{t-1})$. There are two cases, depending on the location of $\mathcal{E}_t$.

Suppose that $y(\mathcal{P}_{t-1}) > y(\mathcal{E}_t)$, so that $\mathcal{E}$ has moved from the first quadrant of $\mathcal{P}$ to her third quadrant, see Figure 2.7(a). Consider the triangle $\mathcal{E}_{t-1}\mathcal{P}_{t-1}\mathcal{E}_t$, which might intersect the boundary of Q. The angle $\angle\mathcal{E}_{t-1}\mathcal{P}_{t-1}\mathcal{E}_t$ is obtuse, which means that $\overline{\mathcal{E}_{t-1}\mathcal{E}_t}$ is the unique longest side. We know that at least one of $\mathcal{E}_{t-1}, \mathcal{E}_t$ is visible to $\mathcal{P}_{t-1}$. If $\mathcal{P}_{t-1}$ sees $\mathcal{E}_{t-1}$ then $d(\mathcal{P}_{t-1}, \mathcal{E}_{t-1}) < 1$, so the game was over at time $t-1$. Therefore $\mathcal{P}_{t-1}$ must see $\mathcal{E}_t$ and $d(\mathcal{P}_{t-1}, \mathcal{E}_t) < 1$, so the game is over after this evader move.

Next, suppose that $y(\mathcal{P}_{t-1}) \leq y(\mathcal{E}_t)$, so that $\mathcal{E}$ has moved from the first quadrant of $\mathcal{P}$ to her

Algorithm 3 Monotone Search Strategy

Require: Vertices of Q are $v_1, v_2, \ldots, v_n$ where $x(v_i) \leq x(v_{i+1})$ for $1 \leq i < n-1$.
Require: Π is the search path
Require: $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_{t-1})$, but $\mathcal{E}_{t-1}$ might be invisible to $\mathcal{P}_{t-1}$
while NOT $\big(\mathcal{E}_{t-1}$ is visible and $0 \leq x(\mathcal{E}_{t-1}) - x(\mathcal{P}_{t-1}) \leq 1/2\big)$ **do**
    Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
    Set checkpoint $c \in \partial Q \cap X(\mathcal{P}_{t-1})$ to be the point nearest to $\mathcal{P}_{t-1}$ such that $x(c) \leq x(\mathcal{P}_{t-1})$
    if $\mathcal{E}_t$ is visible and $d(\mathcal{P}_{t-1}, \mathcal{E}_t) \leq 1$ **then**
        $\mathcal{P}_t \leftarrow \mathcal{E}_t$, which captures the evader
    else if $-1/2 \leq x(\mathcal{E}_t) - x(\mathcal{P}_{t-1}) < 0$ **then**
        Note: $\mathcal{E}_t$ is in evader territory and visible by Lemma 2.1
        $\mathcal{P}'_t \leftarrow (x(\mathcal{E}_t), y(\mathcal{P}_{t-1}))$
    else
        Note: $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_t)$
        $k \leftarrow$ the unique index for which $x(v_k) \leq x(\mathcal{P}_{t-1}) < x(v_{k+1})$
        $Z \leftarrow$ the rightmost point in $\Pi \cap B(\mathcal{P}_{t-1}, 1)$
        $\mathcal{P}_t \leftarrow$ the unique point on Π with x-coordinate $\min\{x(v_{k+1}), x(Z), x(\mathcal{E}_t) - 1/2\}$
    end if
    Update $t \leftarrow t+1$
end while

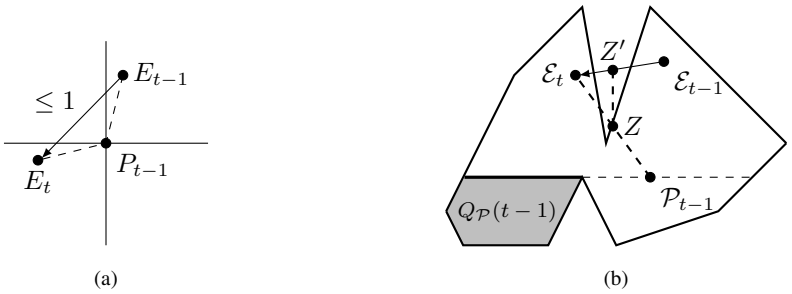
second quadrant, see Figure 2.7(b). Assume that $\mathcal{P}_{t-1}$ cannot see $\mathcal{E}_t$. Since we are following the search strategy, no lower feature to the left of the pursuer can impede her horizontal movement. Therefore the pursuer's visibility must be blocked by the upper chain Π_U. In other words, $\overline{\mathcal{P}_{t-1}\mathcal{E}_t}$ intersects Π_U at some point Z; see Figure 2.6(a). Let Z' be the point on $\overline{\mathcal{E}_{t-1}\mathcal{E}_t}$ that is directly above Z. Since Z is on the boundary of Q, the point Z' lies outside Q. Therefore $\overline{\mathcal{E}_{t-1}\mathcal{E}_t}$ also intersects Π_U, which means that $\mathcal{E}_t$ was not visible from $\mathcal{E}_{t-1}$, and so the evader could not have moved to $\mathcal{E}_t$. We conclude that $\mathcal{P}_{t-1}$ sees $\mathcal{E}_t$. □

We can now prove that the pursuer achieves rook position by following the search strategy.

Lemma 2.2. *If $\mathcal{P}$ follows the search strategy then $\mathcal{E}$ cannot step into the pursuer territory without being caught. Furthermore, $\mathcal{P}$ either achieves rook position or captures $\mathcal{E}$ within $O(n(Q) + \operatorname{diam}(Q))$ rounds.*

Proof. We show that two loop invariants are maintained by the pursuer: $\mathcal{E}_t \notin Q_{\mathcal{P}}(t)$ and $x(\mathcal{P}_t) \leq x(\mathcal{E}_t)$. We also show that the pursuer territory increases over time, which ensures that search strategy terminates. Initially $\mathcal{P}$ is located at the leftmost vertex $\mathcal{P}_0 = v_1$, so $x(\mathcal{P}_0) \leq x(\mathcal{E}_0)$ and the pursuer territory $Q_{\mathcal{P}}(0) = \{v_1\}$. Now suppose that $\mathcal{P}$ has not established rook position by time $t-1$ with $x(\mathcal{P}_{t-1}) < x(\mathcal{E}_{t-1})$ and $\mathcal{E}_{t-1} \notin Q_{\mathcal{P}}(t-1)$. Since $\mathcal{P}$ is not in rook position, either $x(\mathcal{E}_{t-1}) > x(\mathcal{P}_{t-1}) + 1/2$, or the evader is hidden and $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_{t-1}) \leq x(\mathcal{P}_{t-1}) + 1/2$. At this point, the evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$. There are four cases, three of which result in rook position (or capture).

Case 1: Either $x(\mathcal{P}_{t-1}) + 1/2 < x(\mathcal{E}_t)$, or $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_t)$ and $\mathcal{P}_{t-1}$ does not see $\mathcal{E}_{t-1}$. Note that $\mathcal{E}_t \notin Q_{\mathcal{P}}(t-1)$ since $x(c_{t-1}) \leq x(\mathcal{P}_{t-1}) < x(\mathcal{E}_t)$ and c_{t-1} is the rightmost point in $Q_{\mathcal{P}}(t-1)$.



(a) (b)

Figure 2.7: The evader cannot hide when moving past the pursuer to the left. (a) When $\mathcal{E}$ moves from the first quadrant to the third quadrant, at least one of E_{t-1}, E_t is visible to P_t and within unit distance. (b) When $\mathcal{E}$ starts above $\mathcal{P}$, he cannot hide behind an upper feature.

In response, the pursuer moves forward along Π. If she encounters the x-coordinate of a vertex, encounters the boundary, or achieves $x(\mathcal{E}_t) - x(\mathcal{P}_t) \leq 1/2$ with visibility, then she stops; otherwise she moves unit distance. Either way, she maintains $x(c_t) \leq x(\mathcal{P}_t) < x(\mathcal{E}_t)$ and $\mathcal{E}_t \notin Q_{\mathcal{P}}(t)$. After this move, if $x(\mathcal{E}_t) - x(\mathcal{P}_t) \leq 1/2$ with visibility, then the pursuer has achieved rook position; otherwise we continue in search mode. The number of such moves is $O(\mathrm{diam}(Q) + n)$.

Case 2: $\mathcal{P}_{t-1}$ sees $\mathcal{E}_t$ and $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_t) \leq x(\mathcal{P}_{t-1}) + 1/2$. This means that $\mathcal{P}$ is in already in rook position, so we take $\mathcal{P}_t = \mathcal{P}_{t-1}$ and $c_t = c_{t-1}$. Since $x(c_t) \leq x(\mathcal{P}_t) \leq x(\mathcal{E}_t)$, we have $\mathcal{E}_t \notin Q_{\mathcal{P}}(t)$ (unless $c_t = \mathcal{P}_t = \mathcal{E}_t$, which ends the game).

Case 3: $x(\mathcal{E}_t) < x(\mathcal{P}_{t-1})$ and $\mathcal{E}_t \in Q_{\mathcal{E}}(t-1)$. The evader position $\mathcal{E}_t$ is visible to $\mathcal{P}_{t-1}$ by Lemma 2.1. Since the leftmost point of $Q_{\mathcal{E}}(t-1)$ is the left endpoint of X'_t, the pursuer can move horizontally to the point $(x(\mathcal{E}_t), y(\mathcal{P}_{t-1}))$ to achieve rook position because $x(\mathcal{P}_t) - x(\mathcal{E}_t) \leq x(\mathcal{E}_{t-1}) - x(\mathcal{E}_t) \leq 1$.

Case 4: $x(\mathcal{E}_t) < x(\mathcal{P}_{t-1})$ and $\mathcal{E}_t \in Q_{\mathcal{P}}(t-1)$, so that the evader crossed the guarded frontier X'_{t-1}. This means that $\mathcal{E}_t$ is not visible to $\mathcal{P}_{t-1}$, blocked by the feature containing the current checkpoint. As observed in the proof of Lemma 2.1, this means that $d(\mathcal{P}_{t-1}, \mathcal{E}_{t-1}) \leq 1$ and $\mathcal{E}_{t-1}$ was visible to $\mathcal{P}_{t-1}$. Therefore the game was over at time $t-1$, and this case cannot occur. □

Once $\mathcal{P}$ establishes rook position, we immediately update the guarded frontier to be $X'_t := X_t$ and update the checkpoint c_t to be the righthand endpoint of this segment. To emphasize the fact that $\mathcal{P}$ is in rook position, we refer to X_t as the **rook frontier**. In the next section, we will see how a pursuer can methodically advance the rook frontier via rook's strategy. Any leftward movement of $\mathcal{E}$ can be copied by $\mathcal{P}$ since the leftmost point in $Q_{\mathcal{E}}(t)$ is the endpoint of X_t. The pursuer can also mimic the evader's rightward movement, up to the checkpoint c_t. If the pursuer is impeded by the boundary to the right, she will start a new search mode. Throughout these horizontal moves, the pursuer territory is guarded from incursion.

2.2 Rook Strategy

After searching, $\mathcal{P}$ uses the **monotone rook strategy** of Algorithm 4 to either capture $\mathcal{E}$ or to reduce the evader territory. We adapt of the rook's strategy for pursuit on polyhedral surfaces of Noori and Isler [14, 15] to line-of-sight pursuit in a monotone polygon. For simplicity, we assume that $\mathcal{E}$ starts in the first quadrant of $\mathcal{P}$. The other cases follow analogously by swapping left for right, and/or up for down.

Recall that there are two types of rook position: pocket position and chute position, as shown in Figure 2.4. We will see that pocket position is the endgame of the pursuit: $\mathcal{P}$ can maintain rook position and methodically advance her rook frontier until capture without re-entering search mode. In an upper pocket position, the lower chain Π_L lies beneath the rook frontier $X(\mathcal{P})$, so features on Π_L cannot impede the pursuer's movements. Meanwhile we will see that any attempt by $\mathcal{E}$ to use upper features to his advantage can be immediately neutralized. The analogous statements hold for a lower pocket position. On the other hand, if $\mathcal{P}$ is in chute position, then $\mathcal{E}$ can break out of this rook configuration via an **escape move**, as shown in Figure 2.9: the evader moves rightward so that the boundary blocks the rook response. This forces the pursuer to abandon rook mode and initiate a new search mode. In this transition to a new search mode, we will have increased the pursuer territory, so we have made progress towards eventual capture.

We deal with pocket position first, and take up chute position.

Lemma 2.3. *If $\mathcal{P}$ is in pocket position then the monotone rook strategy captures the evader in $O(\mathrm{area}(Q_{\mathcal{E}}))$ turns.*

The proof requires some helper lemmas, paying careful attention to visibility and mobility constraints. First, we show that the evader cannot approach $X(\mathcal{P})$ or leave $Q_{\mathcal{E}}$ without being captured.

Lemma 2.4. *Suppose $\mathcal{P}$ and $\mathcal{E}$ are in pocket position at time $t-1$. If the evader moves so that $|y(\mathcal{P}_{t-1}) - y(\mathcal{E}_t)| \leq \sqrt{3}/2$, then the pursuer can move to a point $\mathcal{P}_t$ that captures $\mathcal{E}$. In particular, $\mathcal{E}$ cannot cross the rook frontier $X(\mathcal{P}_{t-1})$ without being captured.*

Proof. Rook position means that $\mathcal{P}$ sees $\mathcal{E}$ and that $\Delta x(t-1) \leq 1/2$. Therefore $\Delta y(t-1) > \sqrt{3}/2$, otherwise the evader was captured at time $t-1$. Suppose that the evader does not cross the rook frontier and that $|y(\mathcal{P}_{t-1}) - y(\mathcal{E}_t)| \leq \sqrt{3}/2$. Since $y(\mathcal{E}_t) < y(\mathcal{E}_{t-1})$ and $\mathcal{P}_{t-1}$ can see $\mathcal{E}_{t-1}$, it is clear that $\mathcal{P}_{t-1}$ can also see $\mathcal{E}_t$. We have $|x(\mathcal{P}_{t-1}) - x(\mathcal{E}_t)| \leq \Delta x(t-1) + 1 = 3/2$ and therefore

$$d(\mathcal{P}_{t-1}, \mathcal{E}_t) \leq \sqrt{(3/2)^2 + (\sqrt{3}/2)^2} = \sqrt{3} < 2.$$

The pursuer can move directly towards E_t to achieve $d(\mathcal{E}_t, \mathcal{P}_t) < 1$, which ends the game.

Finally, suppose that the evader moves to a point $\mathcal{E}_t$ below the rook frontier $X(\mathcal{P}_{t-1})$. We must be careful in this case since features on the lower boundary might prevent $\mathcal{P}$ from moving directly towards $\mathcal{E}_t$. Instead, the pursuer moves laterally along the rook frontier to capture the evader. Let Z be the point of intersection of $X(\mathcal{P}_{t-1})$ and $\overline{\mathcal{E}_{t-1}\mathcal{E}_t}$. Since $\Delta y(t-1) > \sqrt{3}/2$, we have

$$d(\mathcal{P}_{t-1}, Z) \leq |x(\mathcal{P}_{t-1}) - x(\mathcal{E}_{t-1})| + |x(\mathcal{E}_{t-1}) - x(Z)| \leq 1/2 + |x(\mathcal{E}_{t-1}) - x(Z)| < 1.$$

Algorithm 4 Monotone Rook Strategy

Require: $0 \leq x(\mathcal{E}_{t-1}) - x(\mathcal{P}_{t-1}) \leq 1/2$ and $\mathcal{P}_{t-1}$ sees $\mathcal{E}_{t-1}$
Require: $Q_{\mathcal{E}}(t-1)$ is bounded above by Π_U and below by the rook frontier $X(\mathcal{P}_{t-1})$
while $\mathcal{E}$ is not captured **do**
    Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
    if $\mathcal{E}$ made an escape move **then**
        Note: $\mathcal{P}$ must have been in chute position
        Exit (in order to start a new searching phase)
    else
        if $\mathcal{E}$ crossed below $X(P_{t-1})$ **then**
            $\mathcal{P}_t \leftarrow$ the point where the evader crossed $X(\mathcal{P}_{t-1})$, capturing $\mathcal{E}$
        else if $\mathcal{E}_t$ is within $\sqrt{3}/2$ of $X(\mathcal{P}_{t-1})$ **then**
            Note: $\mathcal{E}_t$ must be visible to $\mathcal{P}_{t-1}$
            $\mathcal{P}_t \leftarrow$ the point on $\overline{\mathcal{P}_{t-1}\mathcal{E}_t}$ as close to $\mathcal{E}_t$ as possible, capturing $\mathcal{E}$
        else if $\mathcal{E}_t$ is not in sight **then**
            $\mathcal{P}_t \leftarrow$ the highest reachable point below $\mathcal{E}_{t-1}$
            Note: this advances the rook frontier by at least $\sqrt{3}/2$.
        else if $|x(\mathcal{P}_{t-1}) - x(\mathcal{E}_t)| \leq 1$ **then**
            $\alpha \leftarrow$ the closer of $x(\mathcal{E}_t) \pm 1/2$ to $x(\mathcal{P}_{t-1})$
            $\mathcal{P}_t \leftarrow$ the highest reachable point on vertical line $x = \alpha$
            Note: this advances the rook frontier by at least $7/22$.
        else
            $\mathcal{P}_t \leftarrow$ the highest reachable point on vertical line $x = x(\mathcal{E}_t) - 1/2$
        end if
    end if
    Update $t \leftarrow t + 1$
end while

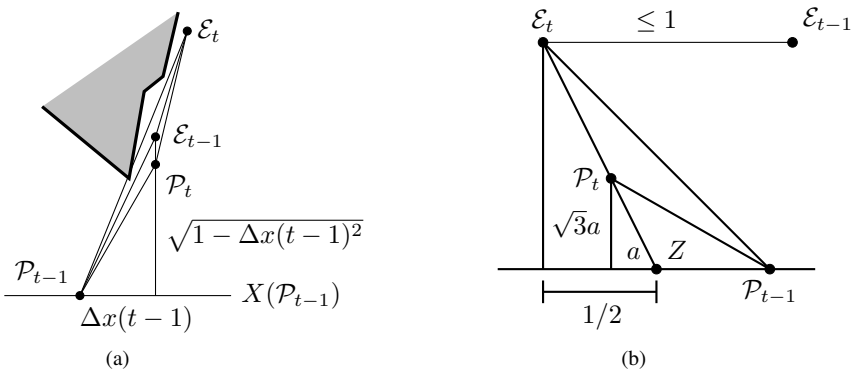


Figure 2.8: Counter-moves during rook mode. (a) If $\mathcal{E}$ hides behind a feature on Π_U then $\mathcal{P}$ responds by moving to a point directly below the evader's previous position. (b) The pursuer response when $\mathcal{E}$ moves leftward past $\mathcal{P}$ and $|x(\mathcal{E}_t) - x(\mathcal{P}_{t-1})| \leq 1$.

Therefore $\mathcal{P}$ can move to Z, and clearly $d(Z, \mathcal{E}_t) < d(\mathcal{E}_{t-1}, \mathcal{E}_t) \leq 1$, so the evader is captured. □

In addition to closing the distance to the evader, our rook strategy maintains (or re-establishes) visibility with the evader at each pursuer turn. The next lemma handles the case in which $\mathcal{E}$ moves to a position that is invisible to $\mathcal{P}$.

Lemma 2.5. *Suppose $\mathcal{P}$ and $\mathcal{E}$ are in pocket position at time $t-1$. If the evader moves to $\mathcal{E}_t$ that is invisible to $\mathcal{P}_{t-1}$ then the pursuer can move to $\mathcal{P}_t$ to re-establish rook position. This move advances the rook frontier by at least $\sqrt{3}/2$.*

Proof. Suppose that $0 \leq \Delta x(t-1) \leq 1/2$ with $\overline{\mathcal{P}_{t-1}\mathcal{E}_{t-1}} \subset Q$, while Π_U obstructs $\overline{\mathcal{P}_t\mathcal{E}_t}$. By Lemma 2.1, $\mathcal{E}_t$ must be to the right of $\mathcal{P}_{t-1}$. We show that the pursuer can move to $\mathcal{P}_t = \mathcal{P}_{t-1} + \left(\Delta x(t-1), \sqrt{1-\Delta x(t-1)^2}\right)$ and has visibility to $\mathcal{E}_t$. We have $x(\mathcal{P}_t) = x(\mathcal{E}_{t-1})$ by construction and $y(\mathcal{P}_t) < y(\mathcal{E}_{t-1})$ (otherwise $\mathcal{E}$ was caught at time $t-1$). Since $\overline{\mathcal{P}_{t-1}\mathcal{E}_{t-1}} \subset Q$ and $X(\mathcal{P}_{t-1}) \subset Q$, we clearly have $\overline{\mathcal{P}_{t-1}\mathcal{P}_t} \subset Q$, see Figure 2.8 (a). Similarly, since $\overline{\mathcal{E}_{t-1}\mathcal{E}_t} \in Q$, we also have $\overline{\mathcal{P}_t\mathcal{E}_t} \in Q$, so the point $\mathcal{P}_t$ sees the point $\mathcal{E}_t$. The pursuer's progress is $\sqrt{1-\Delta x(t-1)^2} \geq \sqrt{3}/2$ since $\Delta x(t-1) \leq 1/2$.

Finally, we show that $\mathcal{P}_t$ is in rook position. Since $x(\mathcal{P}_t) = x(\mathcal{E}_{t-1})$, it is sufficient to prove that $|x(\mathcal{E}_t) - x(\mathcal{E}_{t-1})| \leq 1/2$. Suppose that the evader moves leftward. By Lemma 2.1, we have $x(\mathcal{P}_{t-1}) < x(\mathcal{E}_t)$ and therefore $|x(\mathcal{E}_t) - x(\mathcal{E}_{t-1})| < |x(\mathcal{P}_{t-1}) - x(\mathcal{E}_{t-1})| \leq 1/2$. Suppose that the evader moves rightward. We claim that the slope of $\overline{\mathcal{E}_{t-1}\mathcal{E}_t}$ must be greater than $\sqrt{3}$. Since $\mathcal{E}_t$ is not visible to $\mathcal{P}_{t-1}$, the slope of $\overline{\mathcal{E}_{t-1}\mathcal{E}_t}$ must be strictly larger than the slope of $\overline{\mathcal{P}_{t-1}\mathcal{E}_{t-1}}$, as shown in Figure 2.8 (a). However, $\mathcal{P}$ was in rook position with $\Delta x(t-1) \leq 1/2$ and therefore $\Delta y(t-1) \geq \sqrt{3}/2$, as otherwise the evader was caught at time $t-1$. Therefore the slope of $\overline{\mathcal{P}_{t-1}\mathcal{E}_{t-1}}$ is at least $\sqrt{3}$, so $\Delta x(t) \leq 1/2$.

The argument for $-1/2 \leq \Delta x(t-1) \leq 0$ is entirely analogous. □

Our final lemma shows that the pursuer can advance the rook frontier when $\mathcal{E}$ does not significantly increase his horizontal distance from $\mathcal{P}$.

Lemma 2.6. *Suppose $\mathcal{P}$ and $\mathcal{E}$ are in pocket position at time $t-1$. If the evader moves so that $\mathcal{E}_t$ is visible to $\mathcal{P}_{t-1}$ and $|x(\mathcal{E}_t) - x(\mathcal{P}_{t-1})| \leq 1$ then the pursuer can re-establish rook position while also advancing the rook frontier by at least $7/22$.*

Proof. Without loss of generality, $0 \leq x(\mathcal{E}_{t-1}) - x(\mathcal{P}_{t-1}) \leq 1/2$. Let α be the closer of $x(\mathcal{E}_t) \pm 1/2$ to $x(\mathcal{P}_{t-1})$, and let Z be the point with $x(Z) = \alpha$ that is closest to $\mathcal{P}_{t-1}$, so that $d(Z, \mathcal{P}_{t-1}) \leq 1/2$. Let $\mathcal{P}_t$ be the point on $\overline{Z\mathcal{E}_t}$ at distance one from $\mathcal{P}_{t-1}$. A leftward evader move is shown in Figure 2.8 (b), and the rightward move results in an equivalent situation. The visibility of $\mathcal{E}_t$ from $\mathcal{P}_{t-1}$ ensures that both of $\overline{\mathcal{P}_{t-1}\mathcal{P}_t}$ and $\overline{\mathcal{P}_t\mathcal{E}_t}$ are in Q. By Lemma 2.4, $\mathcal{E}_t$ is at least $\sqrt{3}/2$ above $X(\mathcal{P}_{t-1})$, so the absolute value of the slope of $\overline{Z\mathcal{E}_t}$ is at least $\sqrt{3}$. In addition, $|x(P_{t-1}) - x(Z)| \leq 1/2$ and $|x(\mathcal{E}_t) - x(\mathcal{P}_t)| \leq |x(\mathcal{E}_t) - x(Z)| = 1/2$. Setting $a = |x(P_t) - x(Z)|$, the distance from $\mathcal{P}_t$ to $X(\mathcal{P}_{t-1})$ is at least $\sqrt{3}a$. By the Pythagorean theorem, $1 \leq (1/2 + a)^2 + 3a^2$, or equivalently, $a \geq (\sqrt{13} - 1)/8$. Therefore, the pursuer has advanced the rook frontier by $\sqrt{3}a > 7/22$. □

We are now ready to prove Lemma 2.3.

Proof of Lemma 2.3. For simplicity of notation, we start at $t = 0$. Let s be the length of the rook frontier $X(\mathcal{P}_0)$. We show that after at most $2s$ turns, either the evader is caught or the pursuer advances the rook frontier by at least $7/22$. We repeat this process until the height of the evader territory is at most $\sqrt{3}/2$, so that the evader is caught by Lemma 2.4. The total number of turns to catch the evader is $O(\text{area}(Q_{\mathcal{E}}))$, as explained below.

Let $Q_{\mathcal{E}}$ denote the initial evader territory, bounded by Π_U and $X(\mathcal{P}_t)$. If $s \leq 1$ then $\mathcal{P}$ can move to the midpoint of the frontier on his next turn and then advance the frontier by 1 on all successive turns until catching $\mathcal{E}$. So we consider $s > 1$. Without loss of generality, we assume that $x(\mathcal{P}_0) \leq x(\mathcal{E}_0) \leq x(\mathcal{P}_0) + 1/2$ and $y(E_0) > y(P_0) + \sqrt{3}/2$. We show that after at most $2s$ moves, either the evader is caught or the pursuer advances the rook frontier by at least $7/22$.

Suppose that $\mathcal{P}_{t-1}$ is in rook position with respect to $\mathcal{E}_{t-1}$ and that the evader moves to $\mathcal{E}_t$. There are three cases to consider. First, if $\mathcal{E}_{t-1}$ is not visible to $\mathcal{P}_{t-1}$, then the pursuer moves according to Lemma 2.5, advancing the frontier by $\sqrt{3}/2 > 7/22$. Second, if $|x(\mathcal{E}_t) - x(\mathcal{P}_{t-1})| \leq 1$ then the pursuer moves according to Lemma 2.3, advancing the frontier by at least $7/22$. Third, suppose that $1 < |x(\mathcal{E}_t) - x(\mathcal{P}_{t-1})| \leq 3/2$, which only occurs when $\mathcal{E}$ moves leftward by at least $1/2$. In this case, the pursuer responds by moving to the highest point reachable on the vertical line through $x = x(\mathcal{E}_t) - 1/2$. (If $|x(\mathcal{E}_t) - x(\mathcal{P}_{t-1})| = 3/2$ then this is a purely horizontal move, but otherwise the pursuer makes some vertical progress.) The evader can only make $2s$ such moves before he is forced to make a move covered by the previous two cases. Therefore, after at most $2s$ moves, the pursuer advances the frontier by at least $7/22$.

The worst-case scenario for the capture time is when the evader repeatedly runs the width of the region using horizontal steps of distance 1. This zig-zagging forces the pursuer to trace a path of length $O(\text{area}(Q_{\mathcal{E}}))$. □

We conclude this section with the pursuer response to an escape move from chute position.

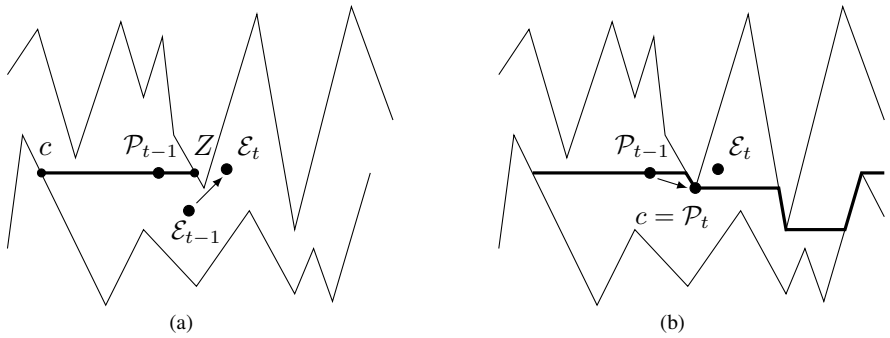


Figure 2.9: Transition from Rook Mode back to Search Mode. (a) $\mathcal{E}$ makes a blocking move, so that $\mathcal{P}$'s responding rook move is obstructed by point Z. (b) Instead, $\mathcal{P}$ counters by re-entering Search Mode with an updated search path and updated checkpoint c.

Lemma 2.7. *Suppose that the pursuer is in lower (or upper) chute position. Using monotone rook strategy, $\mathcal{P}$ can either capture $\mathcal{E}$, or she can update her checkpoint to the right.*

Proof. Suppose that $\mathcal{P}$ is in lower chute position (an upper chute is analogous). If the evader never makes an escape move, then the pursuer will establish a lower pocket position. The pursuer advances the rook frontier, so its right endpoint must eventually transition to the lower chain. In this case, the pursuer catches the evader by Lemma 2.3.

Suppose that $\mathcal{E}$ makes an escape move. That is, $\mathcal{E}$ makes a rightward move below an upper feature that obstructs the pursuer's rook response, see Figure 2.9 (a). Let $Z \in \Pi_L$ be the point that obstructs the pursuer's desired movement. In response, $\mathcal{P}$ re-enters search mode. Her new search path, which is constructed via Algorithm 2, starts from her current location, and then continues rightward, as shown in Figure 2.9 (b). After her first move, she encounters the blocking feature, so her checkpoint c updates to the blocking point Z, or some point on the blocking feature even further to the right. In other words, the entire rook frontier is now guarded. This updates the checkpoint compared to the last time that $\mathcal{P}$ was in search mode. □

2.3 Catching the Evader

We are finally ready to prove Theorem 1.1(a).

Proof of Theorem 1.1(a). The pursuer alternates between Search Mode and Rook Mode until she captures the evader: see Figure 1.2 for an example pursuer trajectory. Once the pursuer is in Search Mode, she will establish rook position by Lemma 2.2. The evader can hide at most n times (once for each vertex of Q), so we switch modes $O(n)$ times. Indeed, $\mathcal{E}$ cannot hide behind the same vertex twice because $\mathcal{P}$ does not exit search mode until $0 \leq x(\mathcal{E}) - x(\mathcal{P}) \leq 1/2$ and $\mathcal{P}$ has visibility of $\mathcal{E}$. Once $\mathcal{P}$ is in Rook Mode, $\mathcal{E}$ must make an escape move in order to hide. Therefore $\mathcal{E}$ must hide behind a new vertex to the right of the previous hiding spot.

Overall, the pursuer spends at most $O(n(Q) + \mathrm{diam}(Q))$ turns in Search Mode: the factor of n appears since $\mathcal{P}$ defensively stops every time she passes the x-coordinate of a vertex of Q, or is blocked by a boundary segment. Note that $\mathcal{P}$ can only pass by a vertex in Search Mode at most once (as any future passings of that vertex must occur in Rook Mode), and each of the n boundary segments can obstruct the pursuer at most once. Therefore the number of truncated steps is $O(n)$.

Each time $\mathcal{P}$ switches to Rook Mode, she either captures $\mathcal{E}$, or $\mathcal{E}$ makes an escape move. In the latter case, the pursuit continues to the right of the current player positions. Crucially, throughout the pursuit, $\mathcal{P}$ never visits the same point twice. Once in Rook Mode, $\mathcal{P}$ guards her pursuer frontier by Lemma 2.4, so the evader cannot cross it. Any vertical move by the pursuer advances this frontier, further limiting the pursuer territory. Furthermore, by Lemma , $\mathcal{P}$ is able to advance the rook frontier by at least $7/22$ once every $\mathrm{diam}(Q)$ moves. Therefore, the entire pursuit path taken by $\mathcal{P}$ has length $O(\mathrm{area}(Q))$.

Every time the pursuer switches from Rook Mode back to Search Mode, she updates her checkpoint and the evader territory shrinks. Therefore, the pursuer ultimately traps the evader in a pocket position and achieves capture by Lemma 2.3 Combining the time bounds for both search and pursuit, the pursuer captures the evader in $O(n(Q) + \mathrm{area}(Q))$ turns. □

This completes our discussion of pursuit in a monotone polygon. In the next two sections, we describe how to adapt this basic algorithm to the scallop and strictly sweepable settings.

3 Pursuit in a Scallop Polygon

A polygon Q is a *scallop polygon* if it can be swept by rotating a line L through some center point C outside the polygon such that the intersection $L \cap Q$ is always convex. An example of a scallop polygon is shown in Figure 1.1 (b). We use polar coordinates $(r : \theta)$ where C is located at the origin, and the polygon is located in the upper half plane. We will sweep our polygon in the clockwise direction, so we index the vertices of our polygon by decreasing angle. More precisely, let the n vertices of Q be $v_k = (r_k : \theta_k)$ for $1 \leq k \leq n$ where $r_k \geq 0$ for all k and and $\pi > \theta_1 > \theta_2 > \cdots > \theta_n \geq 0$. For a point $x \in Q$, let $S(x)$ denote the sweep line through x, and let $\theta(x)$ denote the angle of $S(x)$.

We begin with an overview of the pursuer's strategy. Just as with a monotone polygon, pursuit in a scallop polygon alternates between search mode and rook mode. Scallop search strategy guards a monotone increasing region as the pursuer traverses from left to right (now measured by the decreasing angle of the sweep line). Figure 3.10 (a) shows an example search path. The main difference is how we handle the notions of "horizontal" and "vertical." We change our frame of reference every time we pass a vertex of the polygon, matching the sweeping nature of the polygon itself. The second difference from the monotone case is that only vertices on the lower boundary are taken as checkpoints. The search path can include vertices on Π_U. However, we may have to relinquish these upper boundary points as our frame of reference rotates. Figure 3.10 (b) shows an example of an upper vertex that goes from guarded to unguarded as $\mathcal{P}$ traverses Π down a spoke line. From here forwards, we use the term **auxiliary point** to refer to the transient checkpoints on the upper chain.

The transition between frames must be handled with care. For $1 \leq k \leq n$, let $S_k = S(v_k)$ be the

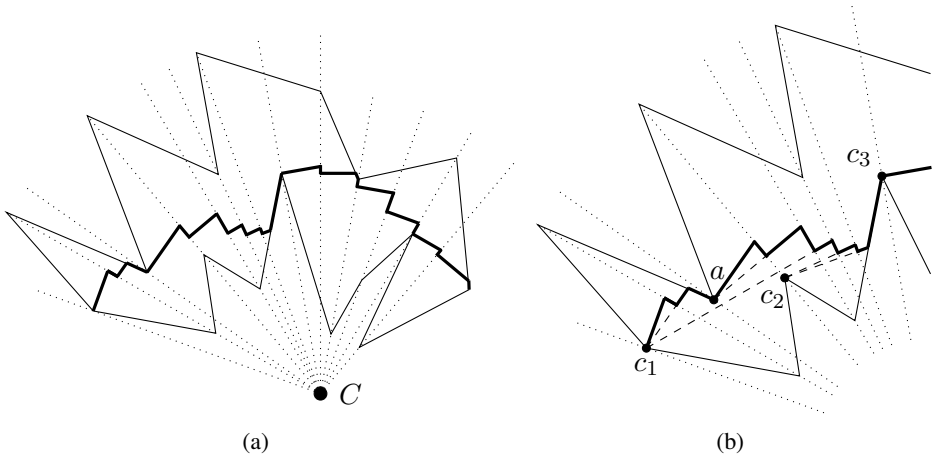


Figure 3.10: The search path of a scallop polygon. (a) At each spoke line, we adjust the path to our new frame of reference so that we continue to guard the checkpoint vertex. (b) Guarding upper boundary vertex a is only temporary: later on, we revert back to guarding c_1 directly. Later on the pursuer updates her checkpoint to c_2, as guarding c_2 also guards the previous checkpoint c_1.

spoke line through the center C and the vertex v_k. When she encounter a spoke line during search, the pursuer halts. She then traverses down the spoke line until she reaches a point that guards the current checkpoint (on the lower boundary) with respect to her new frame of reference. At this point, she adopts that frame of reference, and continues forward. Throughout this transition, she must be aware of both her old and new frames of reference, in case $\mathcal{E}$ tries to invade the pursuer territory.

The goal of searching is to establish rook position in the current frame of reference. The pursuer then transitions into rook mode: she maintains this frame of reference and executes rook strategy for as long as she can maintain rook position. There are now three ways for $\mathcal{E}$ to force $\mathcal{P}$ back into search mode. As in the monotone case, $\mathcal{E}$ can make an escape move to the right, and he now has two additional gambits: he can hide behind a feature, or he can make a blocking move within a pocket, see Figures 3.12 and 3.13 below. Moreover, the return to search mode must be handled with care. To prevent recontamination, the pursuer must keep track of two frames of reference, see Algorithm 8 below.

In the following subsections, we consider scallop search mode, scallop rook mode, and the delicate transitions between then two.

3.1 Search Strategy

Algorithms 5 and 6 adapt the monotone searching of Algorithms 2 and 3 to our new setting. We begin with the result analogous to Lemma 2.2

Lemma 3.1. *If $\mathcal{P}$ follows the scallop search strategy then $\mathcal{E}$ cannot step into the pursuer territory without being caught. Furthermore, $\mathcal{P}$ either achieves rook position or captures $\mathcal{E}$ in finite time.*

Algorithm 5 Create Scallop Search Path

The vertices of Q are $v_1, v_2, \ldots, v_n$, listed in decreasing order of polar angle.
Let spoke line S_i be the line through v_i and C, $1 \leq i \leq n$.
Let transverse line T_i be the line through v_i that is perpendicular to S_i, $1 \leq i \leq n$
Set point $p := v_1$ with vertical axis $Y := S_1$ and horizontal axis $X := T_1$.
Set checkpoint $c := v_1$
while $p \neq v_n$ **do**
    Move p along horizontal axis X until reaching either ∂Q or a spoke line
    if p is on lower boundary Π_L **then**
        Traverse upwards along Π_L until reaching local maximum vertex $x_k \in \Pi_L$
        Set vertical axis $Y := S_k$ and horizontal axis $X := T_k$.
        Update checkpoint $c := p = v_k$
    else
        if p is on upper boundary Π_U **then**
            Traverse downwards along Π_U until reaching reaching local minimum vertex $x_k \in \Pi_U$
        end if
        Set vertical axis $Y := S_k$ and horizontal axis $X := T_k$.
        Move down Y until the leftward horizontal line ℓ through p protects checkpoint c
        if ℓ intersects a vertex $v_j \neq c$ on the lower boundary **then**
            Update checkpoint $c := v_j$, where v_j is the rightmost vertex in $\ell \cap \Pi_L$
        end if
    end if
end while

Algorithm 6 Scallop Search Strategy

Require: Vertices of Q are $O_L = v_0, v_1, \ldots, v_n = O_R$ where $\theta(v_i) \geq \theta(v_{i+1})$ for $1 \leq i < n$.
Require: Π is the search path
Require: Initial frame of reference (X, Y) is the current frame of reference of $\mathcal{P}_{t-1}$
Require: $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_{t-1})$, but $\mathcal{E}_{t-1}$ might be invisible to $\mathcal{P}_{t-1}$

while NOT $\big(\mathcal{E}_{t-1}$ is visible and $0 \leq x(\mathcal{E}_{t-1}) - x(\mathcal{P}_{t-1}) \leq 1/2\big)$ **do**
    Set checkpoint c to be the rightmost guarded point on the search frontier $Y(\mathcal{P}_{t-1})$
    while $\mathcal{P}_{t-1}$ is not on a spoke line **do**
        Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
        $\mathcal{P}_t \leftarrow$ one monotone search move (Algorithm 3) with respect to (X, Y), but stopping if she reaches the next spoke line
        $t \leftarrow t + 1$
    end while
    Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
    $S_k \leftarrow$ the spoke line that $\mathcal{P}$ has encountered
    $(X', Y') \leftarrow$ the frame of reference for S_k
    if $\mathcal{E}_t$ is in the third quadrant of the (X', Y') frame **then**
        $(X, Y) \leftarrow (X'', Y'')$ where $Y'' = \overline{\mathcal{P}\mathcal{E}}$ is the new vertical direction
        Exit to start a new Scallop Rook Strategy (Algorithm 7) with respect to new reference frame
    end if
    $p \leftarrow$ the lowest point on $\Pi \cap S_k$
    while $\mathcal{P}$ has not reached p **do**
        if $\mathcal{E}_t$ is in rook position with respect to frame (X, Y) or frame (X', Y') **then**
            Exit to start a new Scallop Rook Strategy with respect to the appropriate frame
        end if
        $\mathcal{P}_t \leftarrow$ step downwards towards p
        $t \leftarrow t + 1$
        if $\mathcal{P}_{t-1} \neq p$ **then**
            Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
        end if
    end while
    Note: $\mathcal{P}_{t-1} = p$ and $\mathcal{P}$ can now adopt her new reference frame
    $(X, Y) \leftarrow (X', Y')$
end while

Proof. As the pursuer travels between spoke lines S_k and S_{k+1}, the proof is unchanged from that of Lemma 2.2. Indeed, the polygon region between the spoke lines S_{k-1} and S_{k+1} is monotone with respect to the horizontal axis (perpendicular to spoke S_k). Furthermore, the guarded vertex remains guarded while $\mathcal{P}$ moves through this region. We therefore need only prove that $\mathcal{E}$ cannot step into the pursuer territory during the change of frame that occurs at spoke lines.

Suppose that $\mathcal{P}$ reaches spoke line S_{k+1} at point Z_1: an example is shown in Figure 3.11. With respect to the polar coordinates centered at C, the search path takes $\mathcal{P}$ from her current position $Z_1 = (\rho_1 : \theta)$ to $Z_2 = (\rho_2 : \theta)$ where $\rho_1 > \rho_2$. The point Z_2 is chosen so that x_1 is **protected** by the horizontal line through Z_2 with respect to vertical S_{k+1}. The simplest case of protecting x_1 is when this horizontal line intersects x_1. However a feature on the lower boundary might obstruct the line through x_1; in this case, Z_2 is chosen so that the new horizontal line intersects the vertex on the top of this obstructing feature.

Before proceeding along the search path, $\mathcal{P}$ first checks whether a simple change of frame will establish rook position. Let $Y_{k+1} = S_{k+1}$ and let Y_{k+1}^- and Y_{k+1}^+ denote the lines located $\pm 1/2$ from S_{k+1}. Let Y_k denote the line through Z_1 parallel to spoke line S_k, and define Y_k^-, Y_k^+ similarly. Since $\mathcal{P}$ has not established rook position, $\mathcal{E}$ must be to the right of Y_k^+. However, $\mathcal{E}$ could be to the left of Y_{k+1}^- (in region $\mathcal{A}$ of Figure 3.11). In this case, $\mathcal{E}$ must be visible to $\mathcal{P}$ since there are no vertices between x_k and x_{k+1}. Therefore $\mathcal{P}$ can change her vertical axis to be the line through $\mathcal{P}$ and $\mathcal{E}$. This protects the checkpoint vertex c (due to the construction of the search path Π), and establishes rook position.

Otherwise, $\mathcal{E}$ is located to the right of both Y_k^+ and Y_{k+1}^+ (in region $\mathcal{B}$ of Figure 3.11). The pursuer moves down the segment $\overline{Z_1 Z_2}$, keeping track of *two* coordinate systems: she uses Y_{k+1} above her position and Y_k below her position. If the evader steps within $1/2$ distance to either of these lines, then $\mathcal{P}$ responds by establishing rook position with respect to the appropriate frame. Once again, this protects the checkpoint vertex c. Finally, if $\mathcal{P}$ reaches Z_2 without establishing rook position, she commits to the Y_{k+1}-frame (which is also the S_{k+1} frame). If her current search frontier intersects any lower boundary vertices to the left, then $\mathcal{P}$ updates her checkpoint or auxiliary point, if necessary. (In Figure 3.11, we continue to guard auxiliary point x_k.) Having adopted her new frame of reference, she continues her search. □

3.2 Rook Strategy

The eight different rook positions in a scallop polygon are the same as those in Figure 2.3. Algorithm 7 lists the pursuer's rook strategy for upper pockets and upper chutes. Lower pockets and lower chutes are handled similarly, and we discuss these adaptations at the end of this section. Rook pursuit in a scallop polygon is more challenging than the monotone case because the evader has some additional gambits. The evader can make a **hiding move** to disrupt the pursuer's visibility (Figure 3.12(a)) or make a **blocking move** in which the boundary physically impedes the pursuer (Figure 3.13(a)). These moves interrupt the pursuer's rook position, so we need a recovery phase to handle these setbacks. This recovery uses the modified search algorithm listed in Algorithm 8 which remains aware of the

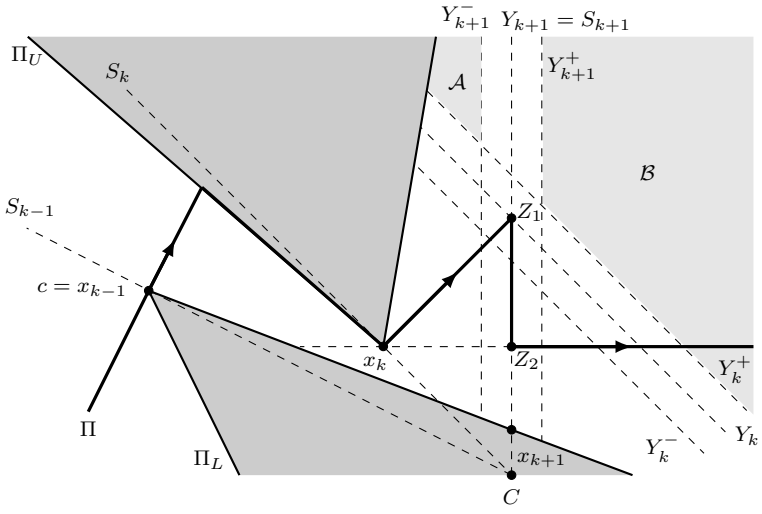


Figure 3.11: Changing frame during the search. The pursuer starts at z_1. If $\mathcal{E} \in \mathcal{A}$, then $\mathcal{P}$ can pick a vertical axis between Y_k and Y_{k+1} and immediately establish rook position. Otherwise, $\mathcal{E} \in \mathcal{B}$, and $\mathcal{P}$ can travel from z_1 to z_2 and safely transition from vertical axis Y_k to vertical axis Y_{k+1} (or establish rook position along the way if $\mathcal{E}$ makes an unwise move).

rook frame of reference, even as the search frame rotates. After recovery in a pocket, $\mathcal{P}$ will return to using rook moves, perhaps with an updated frame of reference.

Recovery in a lower (upper) chute must deal with the case where $\mathcal{E}$ makes an **escape move** using the upper (lower) boundary to block the pursuer's rightward movement (Figures 3.15 and 3.14). In response to such an escape move, the pursuer will either re-establish a rook position, or initiate a new search phase. Once again, the transition in a scallop polygon is more difficult than in a monotone polygon, due to the shifting frame of reference.

We will consider upper pocket position first.

Lemma 3.2. *If $\mathcal{P}$ and $\mathcal{E}$ are in upper pocket position, then upper rook strategy captures the evader in $O(\mathrm{area}(Q_\mathcal{E}) + n(Q_\mathcal{E}))$ turns, where $n(Q_\mathcal{E})$ is the number of vertices in the pocket $Q_\mathcal{E}$.*

Proof. As in the proof of Lemma 2.3, we show that $\mathcal{P}$ consistently shrinks the size of the pocket. Furthermore, the pursuer trajectory never visits the same point twice, capturing the evader in $O(\mathrm{area}(Q_\mathcal{E}) + n(Q_\mathcal{E}))$ turns.

Let $S_k = S(v_k)$ be the spoke line perpendicular to the rook frontier. Suppose that $\mathcal{P}$ and $\mathcal{E}$ are in upper pocket position and that $\mathcal{P}$ is executing the strategy of Algorithm 7. If $\mathcal{E}$ never makes a hiding move or a blocking move, then $\mathcal{P}$ captures $\mathcal{E}$ in $O(\mathrm{area}(Q_\mathcal{E}))$ turns by the same argument used for Lemma 2.3. Suppose that $\mathcal{E}$ hides behind a feature in the upper pocket. We consider the case when $x(\mathcal{P}_{t-1}) - 1/2 \leq x(\mathcal{E}_{t-1}) \leq x(\mathcal{P}_{t-1})$ just before this hiding move; the case $x(\mathcal{P}_{t-1}) \leq x(\mathcal{E}_{t-1}) \leq x(\mathcal{P}_{t-1}) + 1/2$ is argued similarly, swapping the roles of left and right.

Let v_ℓ be the **hiding vertex** that obscures the evader. First, we consider the case $\ell \leq k$, see Figure

Algorithm 7 Upper Rook Strategy (for Upper Pocket and Upper Chute)

Require: $|x(\mathcal{P}_{t-1}) - x(\mathcal{E}_{t-1})| \leq 1/2$ and $y(\mathcal{P}_{t-1}) < y(\mathcal{E}_{t-1})$ and $\mathcal{P}_{t-1}$ sees $\mathcal{E}_{t-1}$.

Note: the rook frontier $X(\mathcal{P}_{t-1})$ lies below $\mathcal{E}_{t-1}$
Note: the leftmost endpoint of $Q_{\mathcal{E}_{t-1}}$ is in $\Pi_U \cap X(\mathcal{P}_{t-1})$.
while $\mathcal{E}$ is not captured **do**
    Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
    if $\mathcal{E}$ made a blocking move **then**
        Let $Z \in \Pi_U \cap X(\mathcal{P}_{t-1})$ be the blocking point
        $\mathcal{P}_t \leftarrow Z$, which is reachable in one step
        Enter Cautious Scallop Search Strategy
    else if $\mathcal{E}$ made a hiding move **then**
        Let $v_\ell \in \Pi_U$ be the hiding vertex.
        Note: assume that $\mathcal{E}_t$ is to the left of $\mathcal{P}_{t-1}$. We handle $\mathcal{E}_t$ to the right of $\mathcal{P}_{t-1}$ similarly.
        while $\mathcal{E}$ remains hidden by v_ℓ, meaning $x(\mathcal{E}_t) < x(v_\ell)$ and $y(\mathcal{P}_{t-1}) < y(v_\ell)$ **do**
            if $x(v_\ell) < x(\mathcal{P}_{t-1})$ **then**
                $\mathcal{P}_t \leftarrow$ move leftward towards $x(v_\ell)$ and upwards with remaining movement budget
                Note: it takes at most two rounds to reach $x(\mathcal{P}) = x(v_\ell)$
            else
                $\mathcal{P}_t \leftarrow$ move upwards towards v_ℓ
            end if
            $t \leftarrow t + 1$
            Evader moves from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$
        end while
        if $\mathcal{E}_t$ is still in the hiding pocket and $\mathcal{P}_{t-1} = v_\ell$ **then**
            Enter Cautious Scallop Search Strategy.
        end if
        Note: $\mathcal{E}_t$ stepped to the right of $\mathcal{P}_{t-1}$ and is no longer hidden
        $\mathcal{P}_t \leftarrow$ one monotone rook move (Algorithm 4)
    else if $\mathcal{E}$ made an escape move **then**
        Let $Z \in \Pi_L$ be the blocking point.
        $\mathcal{P}_t \leftarrow Z$, which is reachable in one step
        if $\theta(\mathcal{E}_t) \geq \theta(\mathcal{P}_t)$ **then**
            Update vertical direction to be $\overline{\mathcal{P}_t\mathcal{E}_t}$
            Start a new Upper Rook Strategy with this updated frame of reference
        else
            Create a new search path from $\mathcal{P}_t = Z$ and enter a new Scallop Search Strategy
        end if
    else
        $\mathcal{P}_t \leftarrow$ one monotone rook move (Algorithm 4)
    end if
    $t \leftarrow t + 1$
end while

Algorithm 8 Cautious Scallop Search Strategy

Require: $\mathcal{P}$ starts at a point Z on the boundary
Require: Previous rook frame of reference is (X_0, Y_0)
Require: $\mathcal{E}$ is to the left of $\mathcal{P}$ in frame (X_0, Y_0). A right move is handled analogously.

```
Create a new search path from Z
while E remains to the left of P with respect to previous rook frame (X_0, Y_0) do
    P makes one Scallop Search Strategy move
    if P is in rook position with respect to the search frame then
        Enter a new Upper Rook Strategy
    end if
end while
Note: E has stepped to the right of P with respect to (X_0, Y_0)
Exit (back to the previous Upper Rook Strategy)
```

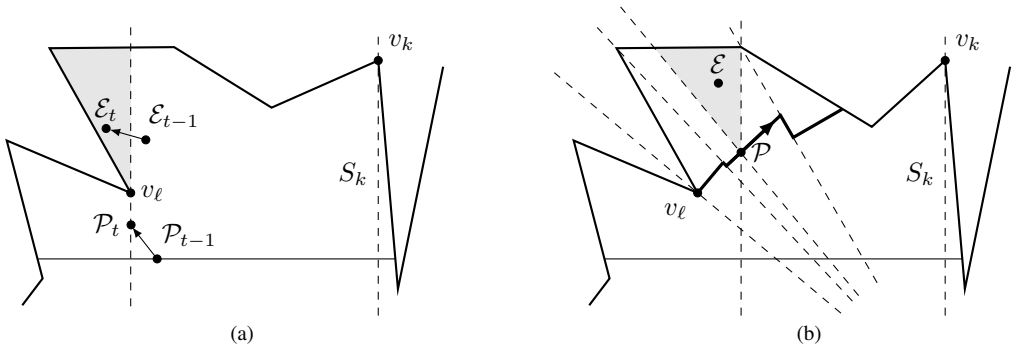


Figure 3.12: (a) $\mathcal{E}$ makes a hiding move into the shaded region above v_ℓ. $\mathcal{P}$ responds by moving below v_ℓ and then upwards until reaching v_ℓ. If $\mathcal{E}$ moves to the right of $\mathcal{P}$ then she re-enters rook mode. (b) Once $\mathcal{P} = v_\ell$, $\mathcal{P}$ starts a new cautious search phase. The evader is confined to the gray region: if $\mathcal{E}$ moves to the right of $\mathcal{P}$ in the original rook frame, then $\mathcal{P}$ reverts to that rook mode.

3.12(a). The evader cannot have stepped to the right of $\mathcal{P}$ by an argument similar to the proof of Lemma 2.1 (with left and right reversed). The pursuer's high-level strategy is to reach the hiding vertex and then traverse a new search path. However, if $\mathcal{E}$ ever moves to the right of $\mathcal{P}$ in the original rook frame, then she reverts to her previous rook strategy. First, $\mathcal{P}$ tries to achieve $x(\mathcal{P}) = x(v_\ell)$, which takes one or two rounds. In the turn that she achieves $x(\mathcal{P}) = x(v_\ell)$, she uses any additional movement budget to move upwards towards the hiding vertex, see Figure 3.12(a). We refer to the region between Π_U and the vertical through v_ℓ as the **hiding pocket**.

After that, the pursuer moves upwards until reaching the hiding vertex v_ℓ. In the meantime, if $\mathcal{E}$ moves to the right of the hiding vertex v_ℓ, then $x(\mathcal{P}) \leq x(\mathcal{E}) \leq x(\mathcal{P}) + 1$ and $\mathcal{E}$ must be visible to $\mathcal{P}$ due to the scallop nature of the polygon. The pursuer responds by returning rook mode using vertical S_k. In particular, she uses a leftward offset, which means that she will make at least $\sqrt{3}/2$ vertical process. This vertical progress is crucial, since it prevents the evader from repeatedly hiding

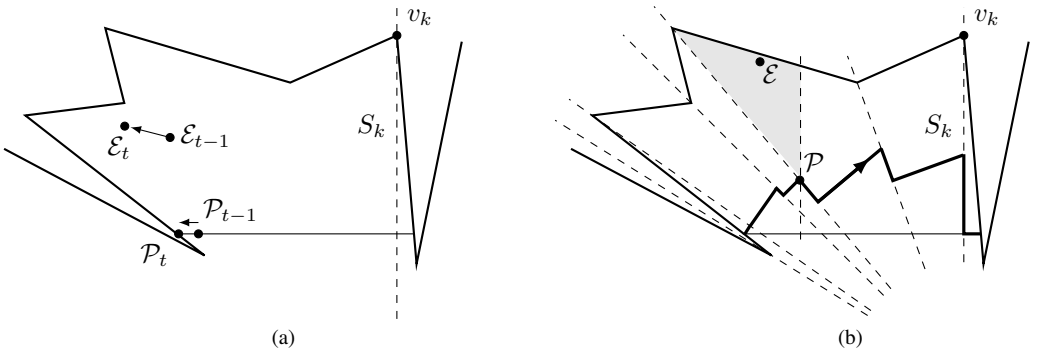


Figure 3.13: (a) The evader makes a blocking move from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$ in an upper pocket. The upper boundary prevents $\mathcal{P}$ from re-establishing rook position, so she moves to blocking point $\mathcal{P}_t$. (b) $\mathcal{P}$ uses the cautious scallop search strategy, while $\mathcal{E}$ is in the shaded region. If $\mathcal{E}$ steps to the right of $\mathcal{P}$ in the original rook frame, then $\mathcal{P}$ will revert back to the old rook phase. $\mathcal{P}$ will achieve rook position before reaching spoke line S_k.

and reappearing above v_ℓ forever. Indeed, once $\mathcal{P}$ reaches v_ℓ, the evader can no longer hide behind this vertex.

Suppose that $\mathcal{E}$ remains in the hiding pocket until $\mathcal{P}$ reaches v_ℓ. The pursuer switches to the cautious scallop search strategy for Algorithm 8, see Figure 3.12(b). She draws a new search path starting at v_ℓ, and searches along this path while keeping track of two frames of reference: her old rook frame (with vertical S_k) and the current search frame. If $\mathcal{E}$ ever steps to the right of $\mathcal{P}$ with respect to the old rook frame, then $\mathcal{P}$ reverts back to the original rook mode. Note that in this case, $\mathcal{P}$ has made vertical progress. Meanwhile, if the evader never steps to the right of $\mathcal{P}$ in the old rook frame, then search mode will terminate with rook position in an updated frame of reference. Furthermore, $\mathcal{E}$ will be trapped in an even smaller pocket than before. Finally, note that $\mathcal{P}$ must attain rook position before reaching spoke line S_k: if $\mathcal{E}$ is to the right of S_k then $\mathcal{E}$ is to the right of $\mathcal{P}$ in the original rook frame, so $\mathcal{P}$ will have already reverted to the previous rook mode.

A hiding move behind a vertex v_ℓ where $\ell > k$ is argued similarly. In this case, it is easy to see that the hiding vertex must satisfy $x(\mathcal{P}_{t-1}) - 1/2 \leq x(v_\ell) \leq x(\mathcal{P}_{t-1}) + 1$. Therefore, $\mathcal{P}$ can achieve $x(\mathcal{P}_t) = x(v_\ell)$ in her first responding move. The argument then proceeds as above, swapping left and right when $x(\mathcal{P}_{t-1}) < x(v_\ell)$.

The pursuer deals with a blocking moving in much the same way. Without loss of generality, suppose that $\mathcal{P}$ is blocked to the right, see Figure 3.13. She moves to the blocking point Z and then draws a new search path from Z. She then enters a modified search strategy, keeping track of both the old rook frame and her current search frame. During the search, if $\mathcal{E}$ steps to the right of $\mathcal{P}$ with respect to the old rook frame, then $\mathcal{P}$ immediately re-enters rook mode with a leftward offset (making $\sqrt{3}/2$ vertical process). Otherwise, $\mathcal{P}$ establishes rook mode with respect to a new frame of reference, and the evader region has been reduced.

During this pursuit, $\mathcal{P}$ never visits the same point twice. She mades methodical progress in each

rook phase. Every time that the evader hides, the pursuer eventually enters another rook phase, and she can only block or hide $n(Q_{\mathcal{E}})$ times. Therefore the capture time is $O(\text{area}(Q_{\mathcal{E}}) + n(Q_{\mathcal{E}}))$. □

When evader territory an a upper (lower) chute, then he can make an escape move, using the lower (upper) chain to obstruct $\mathcal{E}$, see Figure 3.14. As in the monotone case, this is the only evader gambit in an upper (lower) chute that involves the lower (upper) boundary.

Lemma 3.3. *Suppose that $\mathcal{P}$ and $\mathcal{E}$ are in upper chute position. Using the upper rook strategy, $\mathcal{P}$ will either capture $\mathcal{E}$, or make progress on her search path.*

Proof. If $\mathcal{E}$ never makes an escape move, then $\mathcal{P}$ will establish pocket position and capture the evader by Lemma 3.2. Suppose that $\mathcal{E}$ makes a rightward escape move, meaning that the lower boundary blocks the pursuer's rook move. Her recovery phase consists of a single round. The pursuer steps rightward to the blocking point Z, see Figure 3.14. If $S(\mathcal{E})$ is between the vertical axis and $S(\mathcal{P})$, then $\mathcal{P}$ chooses the line between $\mathcal{P}$ and $\mathcal{E}$ as her new vertical axis and starts a new rook phase. This updates the rook frontier and advances her checkpoint. Otherwise, $\mathcal{P}$ starts a new search phase from her current location $z \in \Pi_L$. This protects her previous checkpoint: $\theta(\mathcal{P}) > \theta(\mathcal{E})$ means that $\mathcal{E}$ is to the right of $\mathcal{P}$ in the new search frame. She updates her checkpoint to Z, and begins searching. □

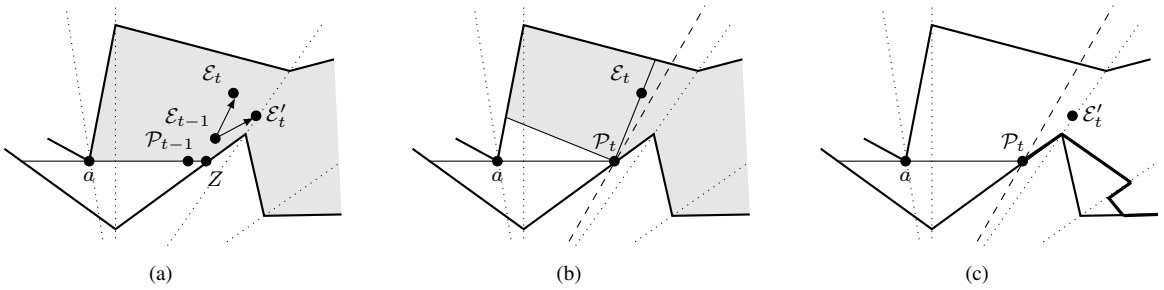


Figure 3.14: Recovery from an escape move in an upper chute. (a) The evader makes one of two rightward blocking moves. The pursuer responds by moving to the blocking point Z. (b) If $\mathcal{E}$ is to the left of the radial line through Z, then $\mathcal{P}$ immediately enters rook mode using the line between $\mathcal{P}$ and $\mathcal{E}$ as the new vertical. (c) If $\mathcal{E}$ is to the right of this line, then $\mathcal{P}$ starts a new search phase.

The lower rook strategy is very similar to Algorithm 7, swapping up for down. Pursuit in a lower pocket is entirely analogous to pursuit in an upper pocket. However, handling an escape move from a lower chute is slightly different. Suppose that $\mathcal{E}$ makes a rightward escape move, meaning that the upper boundary blocks the pursuer's responding rook move, see Figure 3.15. In response, the pursuer steps rightward to the blocking point Z. If she is currently above her most recent search path Π, then she cannot start a new search path from her current location, see Figure 3.11(a). Instead, she travels down the sweep line $S(Z)$ until reaching $Z' \in \Pi \cap S(Z)$. While moving along $S(Z)$, she keeps track of two frames of reference, just as if she is making a usual frame transition during search mode, as shown in Figure 3.11 (c). This protects the last checkpoint of her previous search phase.

We omit the formal adaptation of Algorithm 7 for lower rook strategy. We simply state the lower formulations of our previous two lemmas. The proofs of those lemmas are easily adapted to the lower rook setting.

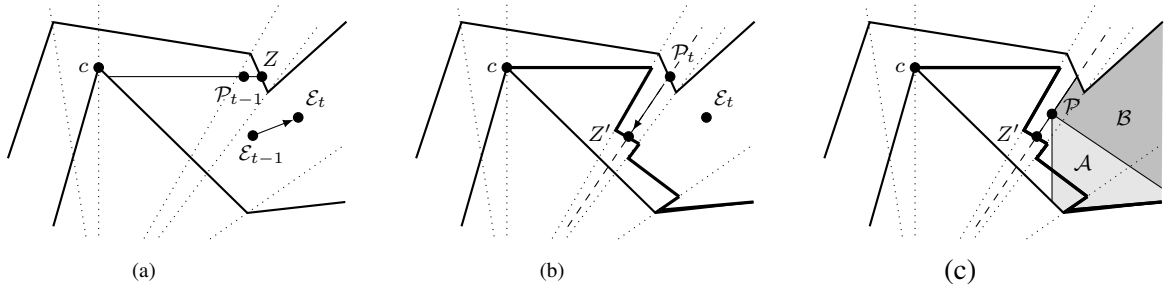


Figure 3.15: Recovery from an escape move in a lower chute. (a) The evader makes an escape move. (b) The pursuer moves to the blocking point Z and then along the sweep line $S(Z)$ heading to point Z' on the previous search path Π. (c) While traversing $S(Z)$, she uses the previous rook frame in region $\mathcal{A}$, and she uses her current frame (where $S(Z)$ is vertical) in $\mathcal{B}$.

Lemma 3.4. *If $\mathcal{P}$ and $\mathcal{E}$ are in lower pocket position, then $\mathcal{P}$ will capture $\mathcal{E}$ in $O(\mathrm{area}(Q_{\mathcal{E}}) + n(Q_{\mathcal{E}})$ moves. If $\mathcal{P}$ and $\mathcal{E}$ are in lower corral position, then, $\mathcal{P}$ will either capture $\mathcal{E}$, or make progress on her search path.* □

3.3 Catching the Evader

The pursuit algorithm for a scallop polygon is identical to Algorithm 1, where we use our scallop algorithms in place of the monotone algorithms. We now prove that this algorithm succeeds in capturing the evader.

Proof of Theorem 1.1(b). The proof is similar to that of Theorem 1.1(a), and relies on Lemmas 3.2 3.3 and 3.4 The pursuer alternates between search mode and rook mode (which includes the cautious search mode). The transition from rook mode to search mode only occurs after an escape move. Every boundary edge or vertex can only be involved in one such escape transition, so we switch modes $O(n)$ times. The search path is at most twice the diameter (by the Pythagorean theorem the search path from O_L to O_R is at most twice the length of the shortest path between these vertices). During search mode, the pursuer never visits the same point twice. Indeed, when $\mathcal{E}$ makes an escape move, $\mathcal{P}$ starts a new search path from the blocking point. The pursuer pauses when passing the spoke lines, accounting for at most n steps. In summary, accounting for all the search modes, the pursuer spends $O(\mathrm{diam}(Q)+n)$ turns searching.

Next, we argue that $\mathcal{P}$ visits each point at most twice. Consider the entire pursuer trajectory from the start until the capture of the evader, and partition the pursuer's path according to each search phase, and rook phase. It is clear that none of the search paths intersect one another. While making rook moves in a fixed rook frame of reference, the pursuer path does not intersect itself. However, it is

possible for the pursuer's path to intersect itself when considering the rook phases before and after an escape move. For an escape move in an upper chute, no point is revisited. When $\mathcal{P}$ is blocked by the lower boundary, she immediately decides whether she can attain rook position via updating her frame of reference. If she can, then this new frame of reference lies above the previous rook frame. Otherwise, she enters search mode, progressing into unexplored territory.

An escape move for a lower chute may lead to $\mathcal{P}$ revisiting some points, but no point will be visited more than twice. Suppose that $\mathcal{E}$ makes an escape move by blocking $\mathcal{P}$ at point $z \in \Pi_U$. The pursuer responds by moving to z and then along $S(z)$ with the goal of reaching the search path Π and transitioning her frame of reference. Consider the turn in which she next attains rook position (either with respect to the old rook frame or with respect to her intended new frame). If $\mathcal{E}$ lies below Π, then she enters rook mode in previously unexplored territory, so no point is revisited. If $\mathcal{P}$ attains rook mode with $\mathcal{E}$ above Π, then the evader territory may include points to the left of $S(z)$ and above Π, see Figure 3.14(c). Therefore this region can be revisited during the new rook phase. However, we are now in upper pocket position or upper chute position, which means that $\mathcal{P}$ will not revisit any points a third time: the next escape move by the evader cannot involve these points in any way.

In a given rook phase, the pursuer makes consistent vertical progress, and she never visits the same point twice. Therefore she spends a total of $O(\mathrm{area}(Q))$ turns in rook mode. Eventually, the evader is trapped in a pocket region, where he is caught by the pursuer. □

4 Strictly Sweepable Polygons

In this final section, we prove Theorem 1.1 (c): a pursuer can capture an evader in a strictly sweepable polygon. Recall that a polygon Q is strictly sweepable when a straight line L can be moved continuously over Q such that (1) $L \cap Q$ is always convex and (2) every point of Q is swept exactly once during this process. This is equivalent to saying that Q can be partitioned into a series of subpolygons $Q_1, Q_2, \ldots, Q_k$ where each Q_i is either a monotone polygon or a scallop polygon, see Figure 4.16. We need to handle four types of transitions: monotone-to-scallop, scallop-to-monotone, scallop-to-scallop where the two centers are on the same side of the polygon, and scallop-to-scallop where the two centers are on opposite sides of the polygon. The previous two sections explain how to handle search and pursuit in each of these component polygons, so our final task is to handle the transition between subpolygons. Rather than presenting formal algorithms, we focus on the alterations required to combine the algorithms for monotone and scallop polygons. Likewise, rather than giving a full proof of Theorem 1.1 (c), we discuss the adaptations needed for the proofs of parts (a) and (b), leaving the details to the reader.

4.1 Search Strategy

First, we address search path construction and search strategy. We have three objectives: (1) define the global search path through the sweepable polygon, (2) manage the search transition from one subpolygon (which requires a change of reference frame), and (3) establish a global notion of search progress as the pursuer moves along the search path.

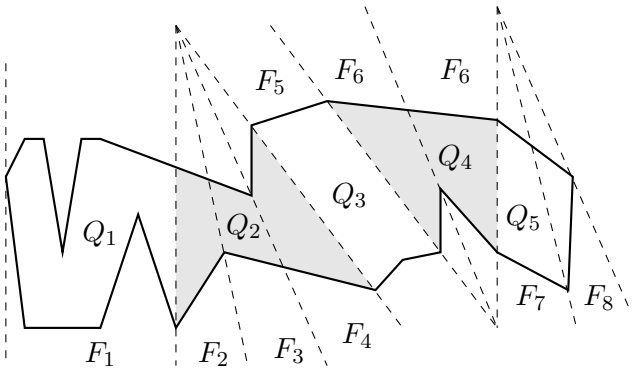


Figure 4.16: A sweepable polygon partitioned by type and by frame of reference. Subpolygons Q_1 and Q_3 are monotone, while Q_2, Q_4, Q_5 are scallop subpolygons. The regions corresponding to the sequential frames of reference are indicated by F_i where $1 \leq i \leq 8$.

We start by constructing the search path, where we must manage the transition from Q_i to Q_{i+1}. For now, we make the simplifying assumption that for $2 \leq j \leq k$, a pursuer in Q_j can see into Q_{j-1}, but features in Q_{j-1} obstruct the view into Q_{j-2}. We will relax this assumption below, once we understand how to handle this basic transition. The simplest transition is from a scallop Q_i to a monotone Q_{i+1}. We treat Q_{i+1} as if it is part of the scallop subpolygon: the construction of Algorithm 5 is still valid. Once the frame of reference for the monotone Q_{i+1} has been established, it protects the last checkpoint in the scallop Q_i. This tactic also handles the transition between scallop polygons whose centers are on the same side of Q.

The transition from a monotone Q_i to a scallop Q_{i+1} is also intuitive. Without loss of generality, assume that the sweeping center is below Q, so that the notions of "up" and "down" are consistent in Q_i and Q_{i+1}. To transition our search path, we view Q_i as part of the scallop subpolygon Q_{i+1}. However, this introduces a new twist that undermines our notion of progress in the monotone Q_i. In a monotone subpolygon, intersections of the search path with either chain establish valid checkpoints. However, in a scallop subpolygon, only intersections with the lower chain are checkpoints: intersections with the upper chain are auxiliary points. If our last checkpoint in monotone Q_i was on the upper chain, then it becomes an auxiliary point once we reach Q_{i+1}. This means that we might have to cede previously guarded area as we move through Q_{i+1}, see Figure 4.17. This compromises the immutability of checkpoints in the monotone Q_i. Therefore, we must introduce a new notion of progress to complement our (now temporary) checkpoint progress.

A simple way to resolve this problem in a monotone-to-scallop transition is to recast the checkpoints on the upper chain as auxiliary points. This is a valid solution when each Q_j does not have visibility into Q_{j-2}: we temporarily give up some progress in Q_i while in Q_{i+1}, but we will recover it by the time we reach Q_{i+2}. However, this solution will not extend to sweepable polygons without this visibility constraint between subpolygons. Indeed, the location of the various sweeping centers invert our notions of "up" and "down," which means that our categorization of a chain being "upper" or "lower" is unstable. Auxiliary points and checkpoints repeatedly switch places as we traverse Q.

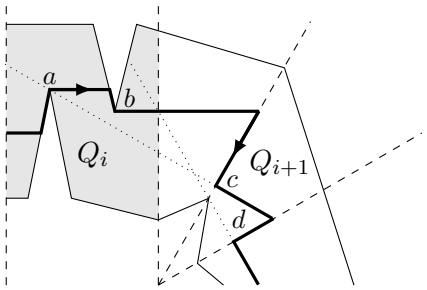


Figure 4.17: Transitioning from monotone Q_i to scallop Q_{i+1}. Points a, b are both checkpoints in monotone Q_i. After transitioning to scallop Q_{i+1}, point b becomes an auxiliary checkpoint. Search path point c protects checkpoint a (relinquishing point b). The checkpoint is updated again when reaching point d.

Therefore, we turn to a new notion of progress that is more robust to changes in reference frames as the pursuer searches the polygon.

To introduce our new progress measure, we return to the global view of our sweepable polygon. Consider a searching pursuer that navigates the entire polygon (using our yet-to-be defined search path). Let us further subdivide the polygon according to the pursuer's frame of reference. In a monotone subpolygon, we use one frame of reference, while in a scallop polygon we use a series of reference frames. Let us denote the entire series of frames for polygon Q as $\mathcal{F} = (F_1, F_2, \ldots, F_\ell)$ where each F_i denotes the notion of horizontal and vertical (particularly, up and down) for the frame, see Figure 4.16. The crucial observation is the following: once $\mathcal{P}$ successfully adopts frame F_i, she never reverts to any of the previous frames $F_1, \ldots, F_{i-1}$. This leads us to our new notion of progress: we say that $\mathcal{P}$ **makes headway** every time that she successfully adopts a new frame of reference. During adoption, she may give up some geometric progress (by abandoning a previous checkpoint), but she has still advanced through the polygon in a global sense. We will show that $\mathcal{P}$ reliably makes headway, so she will eventually establish rook mode. Finally, we note that $|\mathcal{F}|$ is bounded by the number of vertices of Q so there are a finite number of headway phases during the search. Therefore the pursuer will surely establish rook position in finite time.[1]

We handle the monotone-to-scallop transition as follows. We act as if Q_i is part of the scallop polygon Q_{i+1} and use Algorithm 2.2 to continue the search path construction. When searching along this path, rather than measuring progress by advancing checkpoints, $\mathcal{P}$ achieves progress by making headway, which means adopting a new frame of reference that guards some subpolygon in $\cup_{k=1}^{i} Q_i$.

Next, we consider a transition from scallop Q_i to scallop Q_{i+1}, where the sweep centers are on opposite sides of Q. This change of frame reverses the orientation of our vertical axis (meaning "up" and "down" are now flipped). As a consequence, we must swap the roles of checkpoints and auxiliary points in Q_i. Indeed, in our new inverted world, it is now crucial to defend the former auxiliary points in Q_i against incursion (so that they are now treated as checkpoints, to be consistent with our updated

[1]We could have used "making headway" as our measure of progress in a sweepable polygon. However, at that point in the exposition, it was more important to emphasize the complementary roles of checkpoints and auxiliary points.

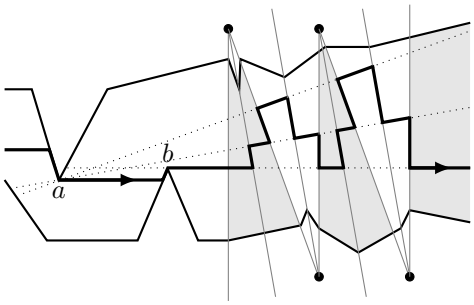


Figure 4.18: A sweepable polygon where the search path alternates between guarding points a and b.

vertical orientation in Q_{i+1}). As in the monotone-to-scallop case, a potential consequence is to cede some previously cleared area in Q_i: our last auxiliary point in Q_i might be "behind" the last checkpoint in Q_i, similar to Figure 4.17. However, we now measure progress by making headway (adopting a new frame of reference) rather than maintaining all previous checkpoints, so we have advanced our global effort.

We claim that the pursuer can always make headway. Indeed, the crucial phases are where the pursuer is traveling along a spoke line in order to transition from frame F_i to frame F_{i+1}. The search path has been constructed so that successful adoption of F_{i+1} guards a previous checkpoint or auxiliary point. Simultaneously, we re-categorize this point as a checkpoint/auxiliary point according to the vertical orientation of the new frame of reference F_{i+1}. By Lemma 5, the pursuer can successfully attain this position or establish rook mode with respect to the previous frame F_i. In the latter case, the last checkpoint (with respect to F_i) remains guarded. This concludes our discussion of the simpler case where Q_j does not have visibility to Q_{j-2}.

Finally, we relax our visibility constraint between successive subpolygons, as promised above. We allow for a point in Q_{i+1} to have visibility to all previous subpolygons $Q_1, \ldots, Q_i$. Our search path construction does not change. However, we must now keep track of all previous checkpoints and auxiliary points in $\cup_{k=1}^{i} Q_k$, since we may revert to guarding any of them during the transition. Figure 4.18 shows a search path that repeatedly alternates between guarding the same two points a, b. We could extend this oscillating behavior $O(n)$ times in the natural way. In this example, as the pursuer advances along the search path, she does not make checkpoint progress. However, she does make headway by incrementing though the list $\mathcal{F}$ of frames of reference. This is our updated notion of progress, so our search strategy terminates in finite time.

4.2 Rook Strategy

Adapting rook's strategy in a sweepable polygon is relatively straight forward. As before, there are two categories of rook position: pocket positions and chute positions. However, the characterization of being an "upper" or "lower" position is no longer appropriate. For example, a pocket position can span a series of subpolygons $Q_i, Q_{i+1}, \ldots, Q_{i+j}$, see Figure 4.19. Therefore, the pursuer responds to any evader gambits (hiding move, blocking move, escape move) by viewing those moves with respect

to the frame of reference (and vertical orientation) appropriate for the current pursuer position.

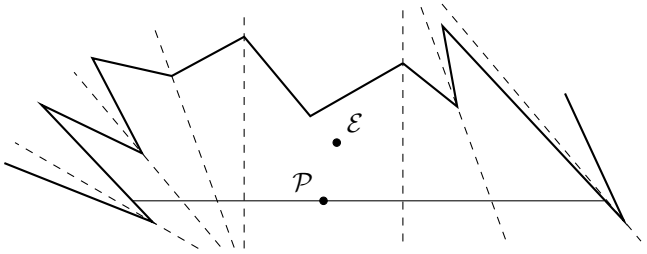


Figure 4.19: A pocket position where the subpolygon types changes. As a result, the pocket transitions from an upper pocket to a monotone pocket to a lower pocket. When $\mathcal{E}$ makes a blocking more or a hiding move, $\mathcal{P}$ responds with the move appropriate to the current subpolygon type.

Likewise, during the pursuer's response to these moves, the subpolygon type might change. Crucially, the smoothness of the sweep line guarantees that the pursuer tactics from Sections 2.1 and 3.1 will re-establish rook position. For example, suppose that the evader makes a leftward blocking move, see Figure 4.20. In response, $\mathcal{P}$ moves horizontally along the rook frontier to the blocking point, and then enters a cautious search phase until either (1) $\mathcal{E}$ steps to the right past $\mathcal{P}$, in which case $\mathcal{P}$ re-enters rook mode using her current reference frame, or (2) she attains rook position with respect to her searching frame. If the polygon type changes during this process, then the search path will continue to protect the blocking vertex. The adaptations for a hiding move and an escape move are similar, so we omit the details.

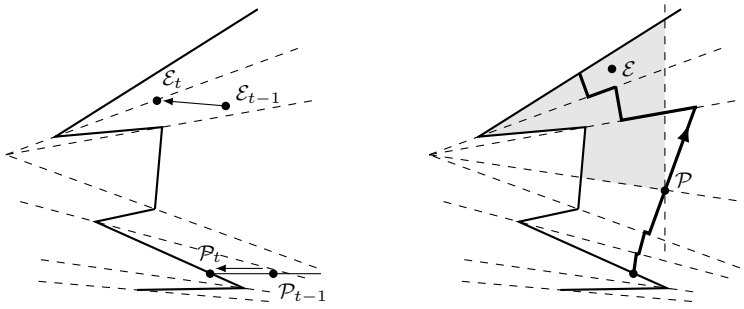


Figure 4.20: A blocking move where the subpolygon type changes within the blocking pocket. (a) $\mathcal{E}$ makes a blocking move and $\mathcal{P}$ moves to the blocking point. (b) $\mathcal{P}$ uses a cautious search strategy, following a new search path that protect the blocking vertex. If $\mathcal{E}$ steps to the right of $\mathcal{P}$ with respect to the previous rook frame, then $\mathcal{P}$ reverts to the previous rook phase.

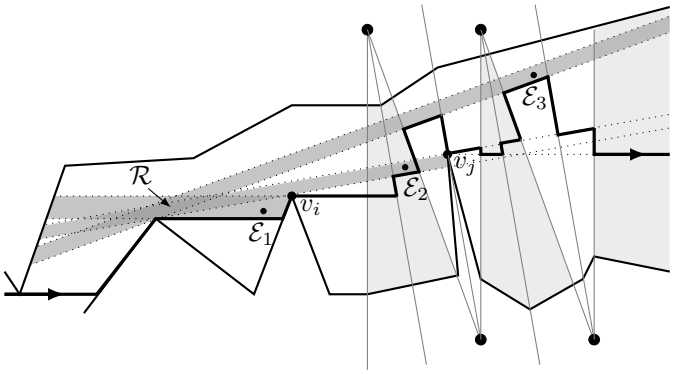


Figure 4.21: The three shaded bands show areas visited in three different rook modes. The rook modes start with the evader at positions $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$, who then moves leftward it is current frame. The first two rook modes end with escape moves at vertices v_i and v_j, respectively. The region $\mathcal{R}$ is visited three times, once in each rook mode.

4.3 Catching the Evader

The pursuit algorithm for a sweepable polygon is identical to Algorithm 1. We conclude by sketching the proof of Theorem 1.1 (c).

Proof of Theorem 1.1(c). The transitions between search phase, rook phase and cautious rook phase are argued just as in the proof of Theorem 1.1(b). Therefore, we only address the capture time bound of $O(n \cdot \text{area}(Q) + \text{diam}(Q))$. As in a scallop polygon, the time spent in search mode is is $O(\text{diam}(Q) + n)$. However, the changing locations of our sweep centers leads to the $n \cdot \text{area}(Q)$ term accounting for rook phases (instead of the $\text{area}(Q)$ term for a scallop polygon). Recall that in scallop pursuit, we visited each point at most twice. Figure 4.21 shows an strictly sweepable example where $\mathcal{P}$ revisits one region $\mathcal{R}$ during three different rook modes; it is straight-forward to extend this example so that one region is revisited $O(n)$ times. Furthermore, simple changes to the geometry (including taking a gentler rotation) can increase the size of the repeated region to be $\Theta(\text{area}(Q))$. Meanwhile, the total number of distinct rook phases is at most $|\mathcal{F}| \leq n$, see Figure 4.16. Due to the repeated visitation during rook mode, the best bound we can guarantee during the active pursuit phases is $O(n \cdot \text{area}(Q))$. This completes our summary of the adaptations required for the proof of Theorem 1.1 (c). □

5 Conclusion

We have considered a pursuit-evasion game in strictly sweepable polygons, including the subfamilies of monotone and scallop polygons. We have shown that a line-of-sight pursuer has a deterministic winning strategy in these environments. Line-of-sight pursuit must alternate between searching and chasing. We have taken advantage of the existence a natural traversal path in our environments, which allows the pursuer to clear the polygon from left to right. In particular, we can associate a unique frame of reference to each point in the polygon by using the sweep line as the vertical axis.

Determining the full class of pursuer-win environments remains a challenging open question. It is known that there are weakly monotone polygons which are evader-win [16]. One milestone would be resolving whether the family of sweepable polygons is pursuer-win. Note that the sweep line is allowed to backtrack as it navigates the polygon. A potential search path must account for this backtracking, which presents a further challenge to the methods presented herein. We are optimistic that our notion of making headway could generalize to all sweepable polygons, but there are certainly technical challenges to overcome. The subfamily of straight walkable polygons would also be a step forward. The boundary of such a polygon can be partitioned into two chains between vertices s and t, such that we can move two mutually visible points monotonically from s to t, one clockwise and the other counterclockwise.

6 Acknowledgments

This work was supported in part by the Institute for Mathematics and its Applications and in part by NSF Grant DMS-1156701. Volkan Isler was supported in part by NSF Grant IIS-0917676. We thank Narges Noori for helpful conversations and feedback.

References

[1] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Appl. Math.*, 8:1–12, 1983.

[2] S. Alexander, R. Bishop, and R. Ghrist. Pursuit and evasion on nonconvex domains of arbitrary dimensions. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[3] S. Alexander, R. Bishop, and R. Ghrist. Capture pursuit games on unbounded domains. *Ensiegn. Math.*, 55:103–125, 2009.

[4] A. Beveridge and Y. Cai. Two-dimensional pursuit-evasion in a compact domain with piecewise analytic boundary. http://arxiv.org/abs/1505.00297.

[5] D. Bhadauria, K. Klein, V. Isler, and S. Suri. Capturing an evader in polygonal environments with obstacles: the full visibility case. *International Journal of Robotics Research*, 31(10):1176–1189, 2012.

[6] A. Bonato and R. Nowakowski. *The Game of Cops and Robbers on Graphs*. American Mathematical Society, 2011.

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

[8] V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion in a polygonal environment. *IEEE Transactions on Robotics*, 5(21):864–875, 2005.

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

[10] K. Klein and S. Suri. Capture bounds for visibility-based pursuit evasion. *Computation Geometry: Theory and Applications*, 48:205–220, 2015.

[11] S. Kopparty and C. V. Ravishankar. A framework for pursuit evasion games in $\mathbb{R}^n$. *Inf. Process. Lett.*, 96:114–122, 2005.

[12] J. E. Littlewood. *Littlewood's Miscellany*. Cambridge Univeristy Press, 1986.

[13] N. Noori, A. Beveridge, and V. Isler. A pursuit-evasion toolkit. Technical Report TR-15-013, University of Minnesota, Computer Science & Engineering Department, 2015.

[14] N. Noori and V. Isler. Lion and man game on convex terrains. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.

[15] N. Noori and V. Isler. Lion and man game on polyhedral surfaces with boundary. In *IEEE Conference on Intelligent Robots and Systems (IROS)*, 2014.

[16] N. Noori and V. Isler. Lion and man with visibility in monotone polygons. *International Journal of Robotics Research*, 33(1):155–181, 2014.

[17] R. Nowakowski and P. Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2–3):235 – 239, 1983.

[18] A. Quilliot. *Jeux et pointes fixes sur les graphes*. PhD thesis, Université de Paris VI, 1978.

[19] J. Sgall. A solution to David Gale's lion and man problem. *Theoretical Comp. Sci.*, 259(1–2):663–670, 2001.