A Complete Pursuit-Evasion Algorithm for Two Pursuers Using Beam Detection

Borislav H. Simov[1], Steven M. LaValle,[2] Giora Slutzki[3]

Abstract

We present an algorithm for a pair of pursuers, each with one rotating beam (flashlight, laser or a camera), searching for an unpredictable, moving target in a 2D environment (simple polygon). Given a polygon with n *edges, the algorithm decides in time* $O(n^4)$ *whether it can be cleared by the pursuers, and if so, constructs a search schedule. The pursuers are allowed to move on the boundary and in the interior of the polygon. They are not required to maintain mutual visibility throughout the pursuit.*

1 Introduction

Consider the following scenario: in a (dark, doorless) polygonal region there are three moving objects (represented as points). Two of them are robots, called the **pursuers** or **1-searchers**, have the task to find the third, called the **evader**. The evader can move arbitrarily fast, and his movements are unpredictable by the pursuers. Each pursuer is equipped with a rotating beam and can see the evader only along the illuminated line segment the beam emits. The pursuers have perfect knowledge about each other's location. They plan their moves in cooperation and are not required to maintain mutual visibility at all times. The pursuers **win** if they illuminate the evader with a beam. If there is a movement strategy of the pursuers whereby they win regardless of the strategy employed by the evader, we say that the polygon can be *cleared by two 1-searchers.* In this paper we present an algorithm which, given a polygon with n edges, decides in time $O(n^4)$ whether it can be cleared by the two 1-searchers, and if so, constructs a search schedule.

The scenario above is a typical problem in pursuit-evasion, a field rising interest in both robotics and computational geometry. The basic task in pursuit-evasion is to compute motion strategies for one or more pursuers to guarantee that unpredictable evaders will be detected. A key difficulty which makes the problem more challenging than basic exploration is that the evaders can sneak back to places already explored. Efficient algorithms that compute these strategies can be embedded in a variety of robotics systems to locate other robots and people. They can aid mobile surveillance systems that detect intruders using sonars, lasers, or cameras. Mobile robots can be used by special forces in high-risk military operations to systematically search a building in enemy territory before it is declared safe for entry.

1.1 Related work

Pursuit-evasion in the plane was introduced by Suzuki and Yamashita [1]. They considered a single pursuer looking for an evader inside a simple polygon. They defined different kinds of pursuers depending on the number of beams (flashlights) that the pursuer is equipped with, e.g., a 1-searcher has one flashlight, a k-searcher has k flashlights, and an ∞-searcher has 360°-vision. This naturally defines a pursuit-evasion problem for each class of searchers. Independently of [1], Icking and Klein [2] defined the "two guard walkability problem", which is a search problem for two guards whose starting and goal position are given, and who move on the boundary of a polygon while maintaining mutual visibility. A solution to the two guards problem was provided in [2], followed by improvements in [3, 4]. While [1, 5, 6, 7, 8] presented polynomial solutions for deciding searchability of special classes of polygons, the general case single pursuer problem was open for quite a while.

Recently, the authors provided a $O(n^3)$ solution for a single 1-searcher in a polygon [9], a result which they later improved to $O(n^2)$ [10]. Park et al [11] presented a polynomial solution for the case of a single 2-searcher and proved that adding more flashlights to a single pursuer does not increase the class of the polygons she can clear. LaValle et al [12] presented a complete algorithm for a single pursuer with 360° vision and extended the pursuit-evasion problem to one in which multiple pursuers collaborate in order to clear a polygonal region. They showed that determining the minimal number of pursuers needed to clear a polygonal region *with holes allowed* is an NP-hard problem. It is not known whether the same problem defined over simple polygons *without holes* is also NP-hard.

Efrat et al [13] considered pursuit-evasion by a chain of k guards as a generalization of the search with two guards. Their pursuit is subject to the restriction that the first and the k-th guards always move on the boundary while

[1]Borislav H. Simov is with the Enterprise Systems Technology Lab, Hewlett-Packard Co., U.S.A., Email: boris_simov@hp.com

[2]Steven M. LaValle is with the Department of Computer Science of the University of Illinois, Urbana, IL, U.S.A., Email: lavalle@cs.uiuc.edu

[3]Giora Slutzki is with the Department of Computer Science of Iowa State University, Ames, IA, U.S.A., Email: slutzki@cs.iastate.edu

guard i, $1 < i < k$ moves in the interior of the polygon and maintains visibility with her neighbors, guards $i-1$ and $i+1$. Efrat et al [13] gave a polynomial algorithm for the k guards problem. Note that the pursuit with two 1-searchers is not a special case of the k guards pursuit since (i) the 1-searchers are not required to maintain visibility all the time, and (ii) for each 1-searcher, the endpoint of the ray of light emitted by her flashlight does not have to move continuously along the boundary of the polygon.

An interesting variation of pursuit-evasion was introduced by Rajko and LaValle [14]. They presented a pursuit-evasion algorithm in which the searcher does not have a map of the polygon. This approach is of special importance for robotics, since it imposes minimal sensing requirements for the pursuer, thus makes a practical implementation more reliable and cost-effective.

1.2 Notation and preliminaries

From now on, all polygons are assumed to be simple. The **boundary** of a polygon P is denoted by ∂P and we assume that $\partial P \subseteq P$ and that ∂P is oriented in the **clockwise** (also called **positive**) direction. For two distinct points $a, c \in \partial P$, we write (a, c) to denote the open interval of all points $b \in \partial P$ such that when starting after a in positive direction along ∂P, b is reached before c. We also use the notation $[a, c]$, $[a, c)$ and $(a, c]$ for the closed and half-closed intervals on ∂P.

Let $p_0, p_1, \ldots, p_{n-1}$ denote the **vertices** on P ordered in the positive direction. The **edges** of ∂P are $e_0, e_1, \ldots, e_{n-1}$, where edge e_i has endpoints p_i and p_{i+1}, and the indices are computed modulo n, e.g., $p_n = p_0$. Vertex $p_i \in \partial P$ is a **reflex vertex** if the angle formed by incident edges e_{i-1} and e_i, in the interior of P, is greater that $180°$ (i.e., points p_{i-1}, p_i, and p_{i+1}, form a left turn). Otherwise, p_i is a **non-reflex vertex**.

We use a standard definition of visibility. For points $c, d \in P$ we say that d is **visible** from c, if every interior point of the line segment $\overline{cd}$ lies in $P - \partial P$. Obviously, if one point is visible from another, then the two are mutually visible. Note that no two points on the same edge of P are mutually visible.

In the rest of the paper "pursuer" will be synonymous to a 1-searcher, unless otherwise specified. Without loss of generality we can assume that the pursuit starts at time 0 and continues until time 1. A region of the polygon which may contain the evader is said to be **contaminated**. Otherwise the region is **clear**.

2 Information-state model of the pursuit

In this section we define a simple model of the pursuit. While the model does not allow feasible search for a winning strategy for the pursuers, it does provide a conceptual framework of the pursuit and is quite intuitive, as it directly represents the movements of the 1-searchers. The idea is that at any time we can represent a snapshot of the pursuit as the position of the pursuers as well as some additional information which records the clear and contaminated regions of the polygon.

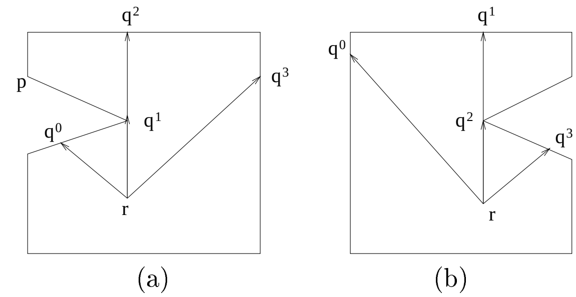


Figure 1: The two kinds of gap edges: (a) left, (b) right.

Let $r \in P$, $q^1, q^2 \in \partial P$ be three colinear points, see Figure 1(a). We say that the pair (q^1, q^2) forms a **left gap edge** relative to r if:

- q^1 is visible from r,
- no point in $(q^1, q^2]$ is visible from r,
- every open interval which contains q^2 also contains a point visible from r.

Similarly, see Figure 1(b), the pair (q^2, q^1) forms a **right gap edge** relative to r if:

- q^2 is visible from r,
- no point in $[q^1, q^2)$ is visible from r,
- every open interval which contains q^1 also contains a point visible from r.

Clearly, the order of q^1 and q^2 is important in the definition of a gap edge. However, from now on we will be a bit more casual about the order, if it can be inferred from the context.

For any two points, $r \in P$, $q \in \partial P$, we refer to the pair $\langle r, q\rangle$ as a **configuration**, if one of the following conditions is true:

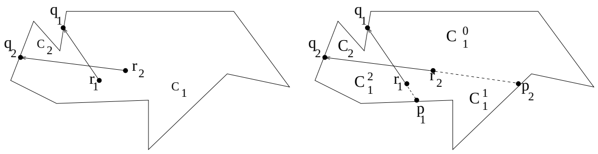
(i) r and q are mutually visible,

(ii) there exists a point $q' \in \partial P$ such that (q', q) forms a left gap edge or (q, q') forms a right gap edge relative to r, or

(iii) $r = q = p_i$ for some non-reflex vertex p_i.

For example, in Figure 1(a), the pairs $\langle r, q^0\rangle$, $\langle r, q^2\rangle$ and $\langle p, p\rangle$ illustrate the three respective cases of the definition of a configuration, while $\langle q^0, r\rangle$ and $\langle r, p\rangle$ are not configurations.

Intuitively, a configuration encodes the coordinates of a single pursuer: r is the location of the pursuer and q is location of her **lightpoint** (the point of the boundary illuminated by her flashlight). While, this is quite natural



(a) (b)

Figure 2: Contamination regions induced: (a) by configurations $\langle r_1, q_1\rangle$ and $\langle r_2, q_2\rangle$ and (b) by the corresponding canonical configurations $\langle p_1, q_1\rangle$ and $\langle p_2, q_2\rangle$.

in case (i) of the definition, how do we interpret cases (ii) and (iii), in which r and q are not mutually visible?

For case (ii) of the definition of a gap edge, note that we can pick a point q'' arbitrarily close to q such that r and q'' will be mutually visible. Thus, while at point r, the pursuer cannot illuminate point q, but she can illuminate points arbitrarily close to q.

Similarly, for case (iii), if $r = q = p$, a pursuer standing at $p' \in \partial P$ can illuminate point $p'' \in \partial P$, such that $p \in (p', p'')$ and p', p'' are arbitrarily close to p.

Consider $q^0, q^1, q^2, q^3 \in \partial P$ ordered in positive direction and $p \in P$. Suppose that p, q^1 and q^2 form a (left or right) gap edge and q^0 and q^3 are sufficiently close to q^1 and q^2 respectively, so that all the points in $(q^0, q^1) \cup (q^2, q^3)$ are visible from p, see Figure 1, (a) or (b). Suppose a pursuer is located at p and the lightpoint is at q^0. If the pursuer rotates the beam clockwise, the lightpoint moves continuously over ∂P before it reaches q^1. At that moment the lightpoint jumps from q^1 to q^2. After q^2 the lightpoint moves continuously to q^3. We call this a **lightpoint jump** from configuration $\langle p, q^1\rangle$ to configuration $\langle p, q^2\rangle$. The reverse move, from $\langle p, q^2\rangle$ to $\langle p, q^1\rangle$, is also a lightpoint jump. We note that a lightpoint jump can be the only possible discontinuity in the location of the lightpoint.

We encode the position of pursuer i with configuration $\langle r_i, q_i\rangle$, $i = 1, 2$. The segments $\overline{r_1 q_1}$ and $\overline{r_2 q_2}$ partition P into a number of connected components. We call each component a **contamination region**, consisting of all points of P which are connected by a path within P not crossed by a ray of light. (To avoid tedious technicalities, we exclude from the partition the points lying on the segments $\overline{r_1 q_1}$ and $\overline{r_2 q_2}$.) For stationary pursuers, an evader can move undetected to every point within a contamination region, hence all the points in a region have the same contamination status. From now on we will simply refer to the region itself as being clear (denoted by 0) or contaminated (denoted by 1). There can be at most 4 contamination regions, so to record their contamination status we need at most 4 bits. We call them **contamination bits** and we write $b \in \mathcal{B}^{14}$ to denote that they can be represented as a nonempty binary string b of length at most 4. Let $s \in \partial P$, $s \neq q_1$ be the point out of r_1, r_2, q_2 which is the first after q_1 in positive direction along ∂P. We label as C_1 the contamination region which borders ∂P in (q_1, s). We label the rest of the regions C_2, C_3, C_4 in positive order along ∂P. Bit b_i represents the status of region C_i, $1 \leq i \leq 4$. For example, in Figure 2(a) there are two contamination regions, C_1 and C_2.

The current status of the pursuit is fully determined by the position of the two pursuers on the boundary and the contamination bits of the regions.

Suppose that $\rho_i = \langle r_i, q_i\rangle$, $i = 1, 2$, are two configurations and b is a binary string of length equal to the number of contamination regions induced by ρ_1 and ρ_2. We define (ρ_1, ρ_2, b) to be an **information state**. The set of all information states is the **information space**.

An information state is called a **starting state** if all contamination bits are 1, i.e., every point of P is contaminated. An information state is called a **goal state** if all contamination bits are 0, i.e., P is clear.

We define a **schedule** to be a piecewise continuous path in the corresponding information space parametrized over time, i.e., we view it as a function $t \to (\rho_1, \rho_2, b)$, encoding the relative position of the two pursuers and the changes in the contamination status over time. More precisely, $\rho_i : [0, 1] \to P \times \partial P$, such that $\rho_i(t) = \langle r_i(t), q_i(t)\rangle$, if at time t pursuer i is located at point $r_i(t) \in P$ and her lightpoint is $q_i(t) \in \partial P$. That is, we will refer to $r_i = r_i(t)$ and $q_i = q_i(t)$ as functions of time as well. A schedule has to also satisfy some conditions derived from the semantics of the pursuit:

- the positions of the pursuers, $r_i(t)$, must be continuous functions of time,
- the endpoints of the ray of light, $q_i(t)$, must be piecewise continuous with discontinuities corresponding exactly to the lightpoint jumps as defined earlier,
- at time 0 the schedule is in a starting information state, and
- the bits in b change according to the contamination status of the respective regions.

A schedule which at time $t = 1$ ends in a goal state is called a **winning schedule**. The definition of a winning schedule directly implies the following lemma.

Lemma 1 *A polygon P can be cleared if and only if a winning schedule exists in the corresponding information space.*

3 Canonical pursuit

In Section 2 we defined a straightforward model of the pursuit, called a schedule. It directly represents the moves of the pursuers, however, its main disadvantage is that the underlying information space is infeasible to search. In this section we will consider a slightly simpler model of the pursuit in which the two 1-searchers stay on the boundary *most of the time*. We will show that the new representation is equivalent to the one from Section 2.

3.1 Canonical configurations

We first define a mapping from a configuration to a corresponding canonical configuration. Suppose a pursuer is at point $r_1 \in P - \partial P$, directing the beam at point $q_1 \in P$, see Figure 2(b). Shoot a ray starting from r_1 in direction opposite of q_1 and let $p_1 \in \partial P$ be the first boundary point hit by the ray. We say that $\langle p_1, q_1 \rangle$ is the **canonical configuration** for the configuration $\langle r_1, q_1 \rangle$. If $r_1 \in \partial P$, then $\langle r_1, q_1 \rangle$ maps to itself. Similarly, define $\langle p_2, q_2 \rangle$ to be the canonical configuration for the second pursuer who is in $\langle r_2, q_2 \rangle$.

The mapping from a configuration to a canonical one can be extended to a mapping from information states to canonical ones. Note that this mapping preserves the position of the lightpoint and has the property that the contamination regions induced by the segments $\overline{p_i q_i}$ are a refinement of the regions induced by the segments $\overline{r_i q_i}$. For example, consider Figure 2(b) and assume the pursuers are in configurations of $\rho_i = \langle r_i, q_i \rangle$, $i = 1, 2$. If we define the area $C_1 = C_1^0 \cup C_1^1 \cup C_1^2$, then $\overline{r_1 q_1}$ and $\overline{r_2 q_2}$ partition P into just two contamination regions, C_1 and C_2, and we can encode the information state as $(\rho_1, \rho_2, b_1 b_2)$. On the other hand, for the corresponding canonical positions, $\pi_i = \langle p_i, q_i \rangle$, $i = 1, 2$, there are four contamination regions: C_2, C_1^0, C_1^1 and C_1^2, and the canonical information state is $(\pi_1, \pi_2, b_2 b_1^0 b_1^1 b_1^2)$. The contamination regions C_1^0, C_1^1, C_1^2 and C_2 are a refinement of the regions C_1 and C_2 which leads us to the following observation.

Observation 2 *At any single moment, we can replace a pair of pursuers in the interior of P with its canonical pair without causing additional contamination.*

The triple $(\langle p_1, q_1 \rangle, \langle p_2, q_2 \rangle, b)$, where $\langle p_1, q_1 \rangle$ and $\langle p_2, q_2 \rangle$ are canonical configurations and b is the corresponding contamination string is defined as a **canonical information state**. A canonical information state is a **starting** (resp. **goal**) one if it is a starting (resp. goal) information state.

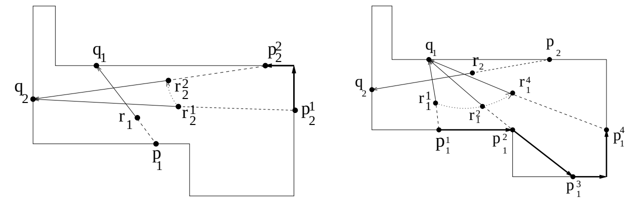
We define the set of all canonical information states to be the **canonical information space**:

$$X = \{(\langle p_1, q_1 \rangle, \langle p_2, q_2 \rangle, b) \mid p_1, q_1, p_2, q_2 \in \partial P, b \in \mathcal{B}^{14}\} \ .$$

3.2 Canonical schedule

Does the mapping from information states to canonical ones directly induce a mapping from a schedule to a canonical schedule? Not always.

Most of the time the continuous motion of $r_i(t) \in P$ translates into a continuous motion of $p_i(t) \in \partial P$, see Figure 3(a). However, there are exceptions: the continuity conditions for π_i may not be satisfied, as seen in the



(a) (b)

Figure 3: Projecting motion in the interior onto motion on the boundary.

example in Figure 3(b). For simplicity, assume that the second pursuer is stationary at point r_2 with lightpoint q_2. The first pursuer moves over a continuous path from r_1^1 to r_1^4, which is projected in a **piecewise continuous** path on ∂P from p_1^1 to p_1^4. The move of the first pursuer can be divided into two parts. The first part, from r_1^1 to r_1^2, is projected into the continuous path from p_1^1 to p_1^2. The second part, from and excluding r_1^2 to r_1^4, is projected into the continuous path from and excluding p_1^3 to p_1^4. However, there is a problem: the jump from p_1^2 to p_1^3 represents a discontinuity in $p_1(t)$, the projection of $r_1(t)$ on ∂P and this jump cannot be simulated by a pursuer moving solely over the boundary.

The solution is to allow the pursuer to move along the segment $\overline{p_1^2 p_1^3}$. Thus the continuous motion from r_1^1 to r_1^4 can be represented as continuous motions from p_1^1 to p_1^2 along ∂P, from p_1^2 to p_1^3 inside P, and from p_1^3 to p_1^4 along ∂P.

We call the move from $\langle p_1^2, q_1 \rangle$ to $\langle p_1^3, q_1 \rangle$ a **pursuer jump**. Just like in the case of the lightpoint jump, the reverse move, from $\langle p_1^3, q_1 \rangle$ to $\langle p_1^2, q_1 \rangle$, is also considered a pursuer jump. Note that in reality the physical location of the pursuer is still continuous but at least for a moment the pursuer had left the boundary. From now on this will be the only circumstance in which the pursuers will leave ∂P.

The introduction of the pursuer jump ensures that there is a well-defined mapping from a schedule to a canonical schedule. Now we can define a **canonical schedule** to be a piecewise continuous trajectory in the corresponding canonical information space parametrized over time. It is quite similar to a schedule with the restriction that the pursuers move on the boundary (except for the pursuer jumps). We can view a canonical schedule as a function $t \to (\pi_1, \pi_2, b)$, encoding the relative position of the two pursuers and the changes in the contamination status over time. More precisely, $\pi_i : [0, 1] \to \partial P \times \partial P$, such that $\pi_i(t) = \langle p_i(t), q_i(t) \rangle$, if at time t pursuer i is located at point $p_i(t) \in \partial P$ and her lightpoint is $q_i(t) \in \partial P$. A canonical schedule has to also satisfy some conditions derived from the semantics of the pursuit:

- The positions of the pursuers, $p_i(t)$, and the lightpoints, $q_i(t)$, must be piecewise continuous functions in ∂P with discontinuities corresponding exactly to the pursuer and lightpoint jumps.

- At time 0 the canonical schedule is in a starting canonical information state.
- The bits in b change according to the contamination status of the corresponding region. For a detailed explanation, refer to Section 3.3.

A canonical schedule which at time 1 ends in a goal state is called a **winning canonical schedule**.
In the rest of this section we show that we do not reduce the power of the pursuers by restricting them to moving on the boundary most of the time. That is, we will show that for a fixed polygon there exists a winning schedule if and only if there exists a winning canonical one.

Lemma 3 *If a given schedule is a winning one, then so is the corresponding canonical schedule.*

Proof: The proof follows from the fact that at any time the canonical schedule induces contamination regions on ∂P which are a refinement of the corresponding regions induced by the original schedule. ∎

Theorem 4 *For a fixed polygon there exists a winning schedule if and only if there exists a winning canonical schedule.*

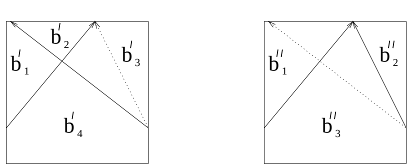
Proof: The reverse direction is trivial, since every canonical schedule is a schedule. The forward direction follows from Lemma 3. ∎

3.3 Changes to the canonical information states

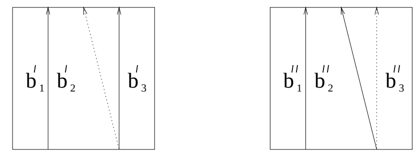
In this section we discuss the ways contamination bits change. This will allow us to define a finite set of elementary moves, so that we regard a canonical schedule as a sequence of these elementary moves.
First, we note an important property of a canonical schedule. The configuration bits only change if there are pursuer of lightpoint jumps or if there is a change in the relative order of the points p_1, q_1, p_2, and q_1 along ∂P. This allows us to divide any canonical schedule into a sequence of moves, called **elementary moves** and defined as follows:

- **type 0: no change**

  Throughout the duration of the move there are no jumps and the order of p_1, q_1, p_2, and q_2 is preserved, so there are no changes in the contamination bits. There are two kinds of type 0 moves depending whether the two beams cross or not.



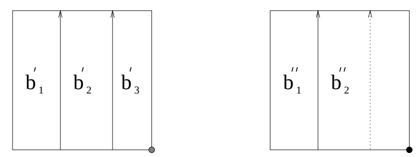
We denote the contamination bits in the left and right positions by b'_i and b''_i, correspondingly, $1 \leq i \leq 4$. A move from the left to the right position does not change the contamination bits, i.e., $b'_i = b''_i$.



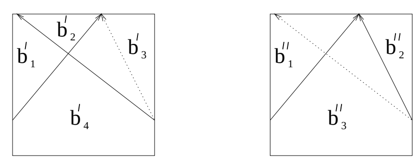
Once again, $b'_i = b''_i$, $1 \leq i \leq 4$.

- **type 1: point merge move**

  The move represents a movement during which the two of the points p_1, q_1, p_2, q_2 merge into a single one. There are two kinds of point merge moves.



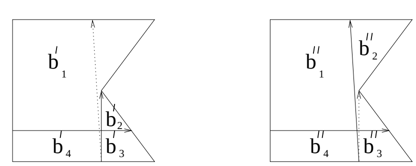
The figure represents a pursuer closing in on a non-reflex vertex. The bits change as follows:
$\Rightarrow$: $b''_1 \leftarrow b'_1$, $b''_2 \leftarrow b'_2$
$\Leftarrow$: $b'_1 \leftarrow b''_1$, $b'_2 \leftarrow b''_2$, $b'_3 \leftarrow 0$



The figure represents crossing of the beams. The bits change as follows.
$\Rightarrow$: $b''_1 \leftarrow b'_1$, $b''_2 \leftarrow b'_3$, $b''_3 \leftarrow b'_4$
$\Leftarrow$: $b'_1 \leftarrow b''_1$, $b'_2 \leftarrow 0$, $b'_3 \leftarrow b''_2$, $b'_4 \leftarrow b''_3$

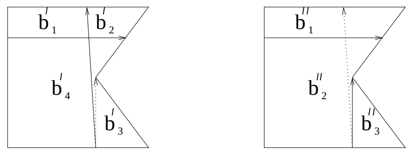
- **type 2: lightpoint jump**

  During this move there is a single lightpoint jump. Both types 2 and type 3 moves have the property that they (at least temporarily) benefit the evader - regions of the polygon that may have be clear before become contaminated. We call this **recontamination**.



The bits change as follows:
$\Rightarrow$: $b''_2 \leftarrow b'_1 \vee b'_2$, $b''_i \leftarrow b'_i$, $i \in \{1,3,4\}$
$\Leftarrow$: $b'_1 \leftarrow b''_1 \vee b''_2$, $b'_i \leftarrow b''_i$, $i \in \{2,3,4\}$

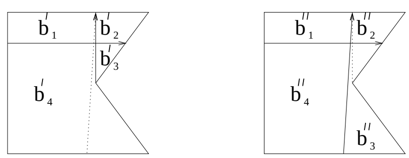
Together with the jump there may also be a simultaneous change in the relative order of the points p_1, q_1, p_2, q_2.



The bits change as follows:
$\Rightarrow$: $b_1'' \leftarrow b_1' \vee b_2'$, $b_2'' \leftarrow b_3' \vee b_4'$, $b_3'' \leftarrow b_3'$
$\Leftarrow$: $b_1' \leftarrow b_2' \leftarrow b_1''$, $b_4' \leftarrow b_2''$, $b_3' \leftarrow b_2'' \vee b_3''$

- **type 3: pursuer jump**

  During this move there is a single pursuer jump, possibly combined with a change of the order of p_1, q_1, p_2, q_2. Just a reminder, eventhough this move suggests a discontinuity of the position of the searcher on ∂P, in reality the position of the pursuer is continuous within P.



The bits change as follows:
$\Rightarrow$: $b_i'' \leftarrow b_i'$, $i \in \{1, 2\}$, $b_3'' = b_4'' \leftarrow b_3' \vee b_4'$
$\Leftarrow$: $b_i' \leftarrow b_i''$, $i \in \{1, 2\}$, $b_3' = b_4' \leftarrow b_3'' \vee b_4''$

We have not listed all the possible moves - there are slight variations depending on the relative order of p_1, q_1, p_2 and q_2. However, we note that the total number of moves is finite, thus it can be written in a lookup table. So in finite amount of time an algorithm can determine a new information state, given a starting information state and a type of move.

4 Finite representation

In the previous section we defined canonical configurations as a simpler model for the two pursuer problem. Yet the canonical information space is still an infinite one, which makes an exhaustive search for a winning schedule infeasible. In this section we will introduce an equivalent, finite representation of the search space and we will show how to find a winning strategy by an exhaustive search in the finite space.

4.1 Inflections and bitangents

Let p be a non-reflex vertex in P. We define $\langle p, p\rangle$ to be a **diagonal configuration**. For example points 10, 13 and 22 are some of the non-reflex vertices in the polygon in

(a)

(b)

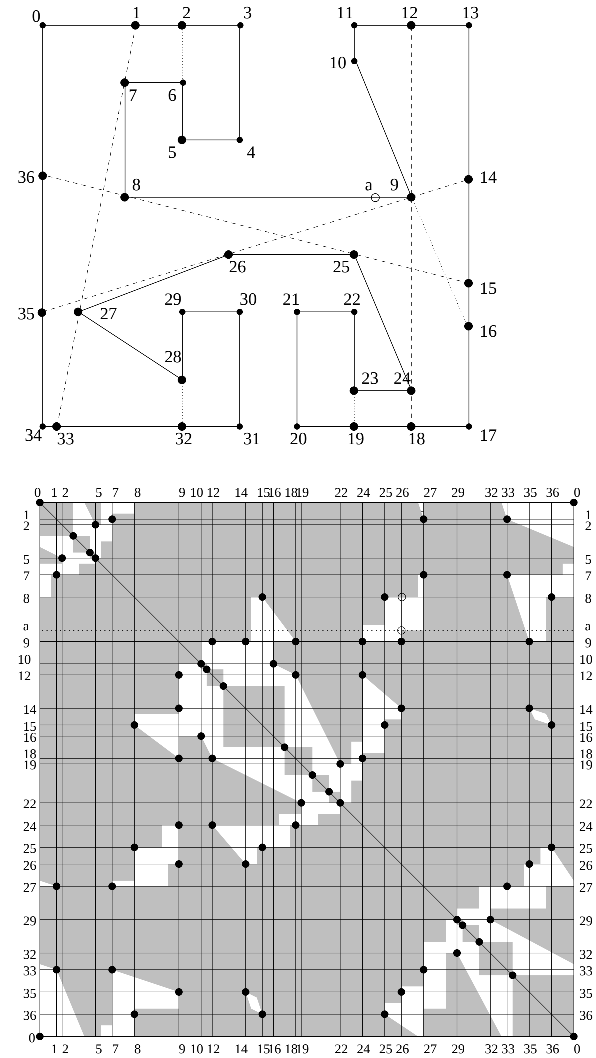


Figure 4: (a) Inflections and bitangents of a polygon. (b) Corresponding canonical configuration space (white areas), diagonal, inflection and bitangent configurations (black circles), boundary configurations (white circles).

Figure 4(a). Also, in the corresponding canonical configuration space in Figure 4(b) the diagonal configurations are marked as black circles along the diagonal, including $\langle 10, 10\rangle$, $\langle 13, 13\rangle$ and $\langle 22, 22\rangle$.

Define an **inflection point** to be a non-reflex vertex, p_i, adjacent to a reflex one, $p_{i\pm1}$. If we shoot a ray from p_i in the direction of the reflex vertex $p_{i\pm1}$, the point $p_i' \in \partial P$ touched first by the ray is defined as a **inflection image**. We define $\langle p_i, p_i'\rangle$ and $\langle p_i', p_i\rangle$ as **inflection configurations**. A sequence of vertices $p_i, p_{i+1}, \ldots, p_{j-1}, p_j$ in which p_i and p_j are the only non-reflex vertices is called a **concave region**. E.g., in Figure 4(a) points 5 and 10 are inflection points, while points 2 and 16 are their corresponding inflection images. There are two concave regions: $5, \ldots, 10$ and $22, \ldots, 29$.

Let $c, d, c', d', x \in P$ be colinear points. Let c and d be mutually visible vertices of ∂P and x be an interior point of $\overline{cd}$. If there exist points c' and d' such that the pairs (c, c') and (d, d') form a gap edges relative to x, we say that c and d define a **bitangent**. We call c and d **bitangent points** and c' and d' **bitangent images**. The polygon in Figure 4(a) has four bitangents, represented by dashed lines colinear with the tuples of points (1, 7, 27, 33), (9, 14, 26, 35), (9, 12, 18, 24) and (8, 15, 25, 36). Let c, d, c' and d' be the bitangent points and projections of some bitangent. If $p, q \in \{c, d, c', d'\}$ and $p \neq q$ we define the pair $\langle p, q\rangle$ as a **bitangent configuration**.

Figure 4(b) represents the Visibility Obstruction Diagram [9] for the polygon in Figure 4(a). The inflection (resp. bitangent) points and images induce red (resp. green) vertical and horizontal lines which form a grid. The white regions represent the set of all canonical configurations. The diagonal, inflection and bitangent configurations are shown as black circles. Consider a single connected region in X. If we traverse its boundary and follow the projections on the horizontal and vertical axes, we observe that the local extremums of the projections coincide with inflection and bitangent points [9]. This prompts us to refer to the inflection and the bitangent configurations as **extreme configurations**.

Consider a vertical red or green (i.e., induced by an inflection or a bitangent) line from the grid. Let $c' = \langle p', q\rangle$ and $c'' = \langle p'', q\rangle$ such that the interior of $\overline{c', c''}$ lies in the interior of X while c' and c'' lie on the boundary of X. We define c' and c'' to be **boundary configurations** with respect to the vertical line $\langle *, q\rangle$. (The definition for a horizontal line is similar.) In Figure 4(b), configurations $\langle 8, 26\rangle$ and $\langle 8, a\rangle$ are configurations with respect to the line $\langle *, q\rangle$.

4.2 Algorithm for finding a search schedule

We now define a directed graph $G = \langle V, E\rangle$ as a finite representation of the canonical information space X. We group the infinite number of canonical information states into equivalence classes, represented by the vertices in the graph.

The vertex set V is defined as follows. For every bitangent configuration $\langle p, q\rangle$ and contamination bit string b:

- for every bitangent configuration $\langle p', q'\rangle$:

$$(\langle p, q\rangle, \langle p', q'\rangle, b) \in V(G)$$

- for every diagonal configuration $\langle r, r\rangle$ (r is a non-reflex vertex in P):

$$(\langle p, q\rangle, \langle r, r\rangle, b), (\langle r, r\rangle, \langle p, q\rangle, b) \in V(G)$$

- for every configuration $\langle r, q\rangle$ (boundary with respect to $\langle p, q\rangle$):

$$(\langle p, q\rangle, \langle r, q\rangle, b), (\langle p, q\rangle, \langle q, r\rangle, b) \in V(G)$$

$$(\langle r, q\rangle, \langle p, q\rangle, b), (\langle q, r\rangle, \langle p, q\rangle, b) \in V(G)$$

Each of the edges of the graph represents an elementary move from one state to another (or from one equivalence class of states to another). The only exception is that in order to keep the outdegree of the graph bounded by a constant, for type 0 moves we only connect configurations along the boundary of X.

A vertex in G is a starting (resp. goal) one, if the corresponding canonical information state is a starting (resp. goal) one. A path in G from a starting to a goal vertex is called a **winning path**.

Theorem 5 *For a polygon P, there exists a winning canonical schedule if and only if there exists a winning path in G.*

So it follows that a polygon can be cleared by two 1-searchers, if and only if there exists a winning path in the graph G.

In order to find a winning path, it suffices to build the graph G and to perform a breadth-first search beginning at a start and ending at a goal vertex. If a winning path exists, then the path represents the search schedule for the two pursuers.

To compute the time complexity of the algorithm above, let n be the number of edges in the polygon and let m denote the number of concave regions, $m = O(n)$. There are 6 bitangent configurations for every bitangent and the number of bitangents is $O(m^2)$, therefore the bitangent configurations are $O(m^2)$. Since the number of inflection configurations is $O(m)$, it follows that the number of extreme configurations is $O(m^2)$ as well. Thus the number of vertices in G is $|V| = O(m^2(n + m^2))$. The outdegree of G is bounded by a constant, so $|E| = O(m^2(n + m^2))$ as well, therefore the time for the search in the graph is $O(m^2(n + m^2)) = O(n^4)$.

5 Conclusion

We presented a complete algorithm for a pair of pursuers, each with one rotating beam, searching for an moving target in a simple polygon. For a polygon with n edges and m concave regions, the algorithm in time $O(m^2(m^2 + n))$ decides whether it can be cleared by the pursuers, and if so, constructs a search schedule. The algorithm can be implemented and embedded on any moving devices with unidirectional vision (flashlights, lasers, or cameras).

A natural direction for extending the current results is designing a similar algorithm for two pursuers with 360° vision. A more ambitious goal is to provide an algorithm for searching a polygon (without holes) using any number of pursuers. Another interesting problem is combining the results of our paper with the minimal sensing approach of Rajko and LaValle [14], i.e., whether the two pursuers can find a winning strategy without prior knowledge of the shape of the polygon.

References

[1] I. Suzuki and M. Yamashita, "Searching for a mobile intruder in a polygonal region," *SIAM Journal on Computing*, vol. 21, no. 5, pp. 863–888, 1992.

[2] C. Icking and R. Klein, "The two guards problem," *International Journal of Computational Geometry and Applications*, vol. 2, no. 3, pp. 257–285, 1992.

[3] P. C. Heffernan, "An optimal algorithm for the two-guard problem," *International Journal of Computational Geometry and Applications*, vol. 6, pp. 15–44, 1996.

[4] L. H. Tseng, P. C. Heffernan, and D. T. Lee, "Two-guard walkability of simple polygons," *International Journal of Computational Geometry and Applications*, vol. 8, no. 1, pp. 85–116, Feb. 1998.

[5] D. Crass, I. Suzuki, and M. Yamashita, "Searching for a mobile intruder in a corridor — the open edge variant of the polygon search problem," *International Journal of Computational Geometry and Applications*, vol. 5, no. 4, pp. 397–412, 1995.

[6] J.-H. Lee, S.-M. Park, and K.-Y. Chwa, "Searching a polygonal room with a door by a 1-searcher," *International Journal of Computational Geometry and Applications*, vol. 8, no. 2, pp. 201–220, 2000.

[7] J.-H. Lee, S. Y. Shin, and K.-Y. Chwa, "Visibility-based pursuit-evasion in a polygonal room with a door," in *Proceedings, ACM Symposium on Computational Geometry (SCG)*, Miami Beach, FL, USA, June 1999, pp. 281–290.

[8] S. M. LaValle and J. E. Hinrichsen, "Visibility-based pursuit-evasion: The case of curved environments," in *Proceedings, IEEE International Conference on Robotics and Automation (ICRA)*, Detroit, MI, USA, May 1999, pp. 1677–1682.

[9] B. H. Simov, G. Slutzki, and S. M. LaValle, "Pursuit-evasion using beam detection," in *Proceedings, IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, USA, Apr. 2000, pp. 1657–1662.

[10] S. M. LaValle, B. H. Simov, and G. Slutzki, "An algorithm for searching a polygonal region with a flashlight," in *Proceedings, ACM Symposium on Computational Geometry (SCG)*, June 2000, pp. 260–269.

[11] S.-M. Park, J.-H. Lee, and K.-Y. Chwa, "Visibility-based pursuit-evasion in a polygonal region by a searcher," Tech. Rep. CS-TR-2001-161, KAIST, 2001.

[12] S. M. LaValle, D. Lin, L. J. Guibas, J.-C. Latombe, and R. Motwani, "Finding an unpredictable target in a workspace with obstacles," in *Proceedings, IEEE International Conference on Robotics and Automation (ICRA)*, 1997, pp. 737–742.

[13] A. Efrat, L. J. Guibas, S. Har-Peled, D. C. Lin, J. S. B. Mitchell, and T. M. Murali, "Sweeping simple polygons with a chain of guards," in *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, San Francisco, CA, USA, 2000, pp. 927–936.

[14] S. Rajko and S. M. LaValle, "A pursuit-evasion BUG algorithm," in *Proceedings, IEEE International Conference on Robotics and Automation (ICRA)*, 2001.