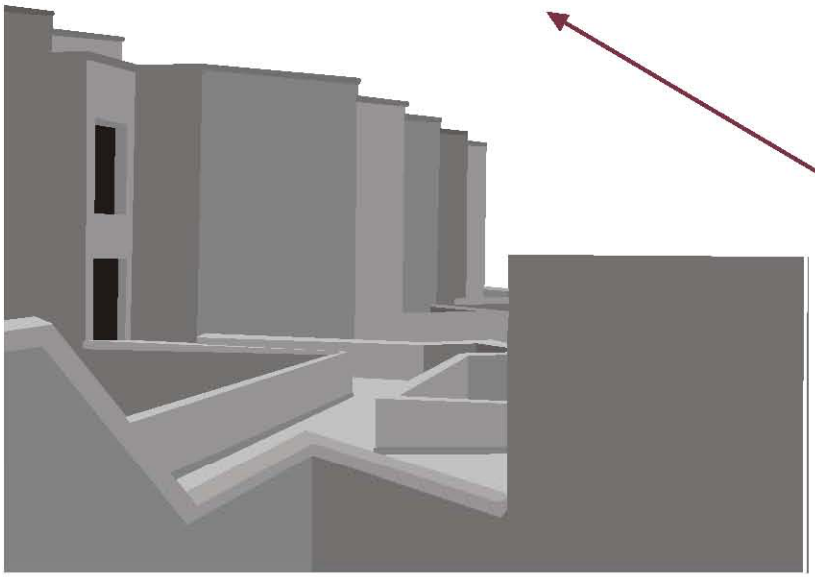

Comunicaciones del CIMAT

EVADER SURVEILLANCE UNDER INCOMPLETE INFORMATION

Mario Santes, Rafael Murrieta-Cid and Raul Monroy

Comunicación del CIMAT No I-09-03/24-02-2009
(CC /CIMAT)

CIMAT

Evader Surveillance under Incomplete Information

Mario Santes†, Rafael Murrieta-Cid† and Raul Monroy‡
†Centro de Investigación en Matemáticas, CIMAT
Guanajuato Mexico
{santes,murrieta}@cimat.mx
‡Tecnológico de Monterrey
Campus Edo. de México
raulm@itesm.mx

***Abstract*— This paper is concerned with determining whether a mobile robot, called the *pursuer*, is up to maintaining visibility of an antagonist agent, called the *evader*. This problem, a variant of *pursuit-evasion*, has been largely studied, following a systematic treatment by increasingly relaxing a number of restrictions.**

In [9], we considered a scenario where the pursuer and the evader move at bound speed, traveling around a known, 2D environment, which might contain obstacles. Then, considering that, in an attempt to escape, the evader travels the shortest path to reach a potential escape region, we provided a decision procedure that determines whether or not the pursuer is up to maintain visibility of the evader and obtained complexity measures of the surveillance task.

In this paper, we prove that there are cases for which an evader may escape only if it does not travel the shortest path to an escapable region. We introduce planning strategies for the movement of the pursuer that keeps track of the evader, even if the evader chooses not to travel the shortest path to an escape region.

We also present a sufficient condition for the evader to escape that does not depend on the initial positions of the players. It can be verified only using the environment.

All our algorithms have been implemented and we show simulation results.

I. Introduction

This paper is to do with pursuit-evasion. We are concerned with determining whether a mobile robot, the *pursuer*, is up to maintaining visibility of an antagonist agent, the *evader*. This problem has been largely studied, following a systematic treatment by increasingly relaxing a number of restrictions.

In a previous paper [9], we considered a scenario where the pursuer and the evader move at bound speed, traveling around a known, 2D environment, which might contain obstacles. Then, considering a simple but appealing escape policy, namely: travel the shortest path to reach a potential escape region, we provided a decision procedure that determines whether or not the pursuer is up to maintain visibility of the evader and obtained complexity measures of the surveillance task.

In this paper, we take a step further: we provide motion planning strategies for a pursuer which has to keep tracking of an evader which does not necessarily follow the shortest path to a escape region. This new setting is of interest, because, as shown in this paper, there exist evasion paths that require the evader *not* to follow this policy. In addition, we assume that the pursuer has no knowledge about the global paths to be taken by the evader, but it knows where the evader will be after a small progress of time.

A. Contributions

This paper makes 4 main contributions:

- We show that if the pursuer does not know the evader motion policy then there are cases where the evader can escape only if it does not travel the shortest distance from its initial position to a escapable region. This result holds regardless of whether the evader is faster or slower than the pursuer.
- We show that determining whether or not a pursuer can maintain visibility of the evader at all times depends on two general factors: (i) The initial positions of both the pursuer and the evader; and (ii) the long-term path plans, that can be executed by the evader.
- We present algorithms that plan pursuer motions so as to keep track of an evader who does not necessarily travel the shortest paths to an escapable region. Our algorithms have been implemented and simulation results are shown.
- We present a sufficient condition for the evader to escape that does not depend on the initial positions of the players and which can be verified using the environment only.

II. Related Work

Keeping track of a moving evader is a popular, long-standing problem, which has been studied from several perspectives. For example, [8] used game theory to approach the problem, yielding an algorithm which attempts to maximize the probability that the evader will remain visible in the future. [5] suggested a method which does not use a global map of the environment; instead, using a local map, built with the help of a range sensor, they run a combinatorial algorithm that computes a differential motion for the pursuer at each iteration.

In [4], the authors presented a local minimum risk function, called the vantage time, used to drive a greedy motion planning strategy.

Others have studied an extended version of the problem involving multiple participants of each kind (evaders and pursuers). [11], for example, developed a method which attempts to minimize the total time in which the evaders escape surveillance. The method, however, was restricted to uncluttered environments. In a similar vein, [7] combined the application of mobile and static sensors. It used a metric for measuring the degree of occlusion, based on the average mean free path of a random line segment.

Pursuit-evasion has be found to be use in interesting applications. For example, in [6], the authors noticed

the similarity between pursuit-evasion games and mobile-routing for networking. Applying this similarity, they proposed motion planning algorithms for robotic routers to maintain connectivity between a mobile user and a base station.

More related to ours is the work of [3], which shows how to efficiently compute a pursuer optimal path in response to a given evader movement. Notice, however, that in [3] the authors want to find the pursuer path associated to *one given evader path*. They do not attempt to deal with the problem of deciding whether or not some (at least one) of all possible evader paths will yield an escaping path.

In this paper, we prove that there are cases for which an evader may escape only if it does not travel the shortest path to an escapable region. Also, we introduce planning strategies for the movement of the pursuer that keeps track of the evader, even if the evader chooses not to travel the shortest path to an escape region (for example, the area behind an obstacle.)

III. Problem definition

The evader and the pursuer are modeled as points moving over a known environment. The environment contains obstacles, each of which is modeled as a polygon. Every participant is assumed to accurately know its position at all times, is equipped with an omni-directional sensor, and is limited to move at bound speed. Other than these, no kinematic nor dynamic constraints are imposed on the pursuer or the evader.

The evader moves continuously and antagonistically. The pursuer does not know the evader motion policy; nor can the pursuer predict it or learn it. However, the pursuer is assumed to know where the evader will be after a small progress of time, Δt. Thus, we assume a universal clock which ticks every Δt units of time; clock ticks are then used to index periods of time. Notice that the pursuer is able to know the whereabouts of the evader, from t to $t + \Delta t$.

Under this setting, we address the problem of discovering pursuer motion strategies that are able to maintain strong mutual visibility of the evader, considering that the global motion policy of it is unknown. Similarly, it is our goal to seek for sufficient conditions, independent of the initial position of the players, such that they guarantee that evader is bound to escape.

A. Strong Mutual Visibility

Let $R_1, \ldots, R_n$ be a partition of the environment, $W = \bigcup_i R_i$, such that each $R_i\,(i \in \{1, \ldots, n\})$ is a convex region. The evader is under pursuer surveillance if strong mutual visibility of the evader by the pursuer holds [9]. Two regions are *strongly mutually visible* if every point belonging to any of the two regions is able to see all the points of the other region. The pursuer maintains strong mutual visibility of the evader, if it is within the same region where the evader is or if they both are in regions that are strongly mutually visible. Thus, maintaining strong mutual visibility of the evader amounts to maintaining visibility of the entire region where it is. This is because strong mutual visibility is stronger than classical visibility.

IV. Preliminaries

A. Environment Partition and Graph Modeling

Region convexity ensures that a robot with omnidirectional sensing is able to see all the points within the region of residence. Our convex partition is similar to the region decomposition produced by the lines of the aspect graph in 2D using perspective projection [2], plus an additional feature, namely: we connect every pair of bitagent vertices. In our partition, bi-tangent rays are extended outward and inward from a pair of bi-tangent points (See Fig. 1.) More details can be found at [9].

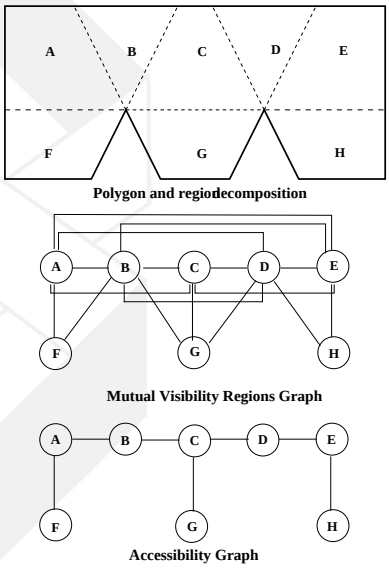

Fig. 1. Environment partition and resulting graphs

B. Two Graphs Modeling the Environment

The partition of the environment yields two graphs, one called *Accessibility Graph* (AG) and the other *mutual visibility graph* (*MVG*). In each graph, nodes represent regions. In an AG, two nodes R_i and R_j are connected, written $(R_i, R_j) \in AG$, if their associated regions share a region boundary bigger than one single point. Likewise, in an MVG, two nodes R_i and R_j are connected, written $(R_i, R_j) \in$ *MVG*, if their associated regions are strongly mutually visible. Using the MVG, each participant is able to know both which regions are candidates to attempt to escape, called *escapable regions*, and which regions the pursuer should move to if an escape is to be prevented, called *prevention-from-escape regions*.

An MVG therefore provides a sufficient condition to maintain evader visibility while an AG defines the possible region transitions that either participant can carry out. Note that what counts as an escapable (respectively prevention-from-escape) region depends on the current regions where both the evader and the pursuer are. More precisely, let E_i (respectively P_j) denote that the evader (respectively

the pursuer) is at region R_i (respectively R_j). For each pair $\langle E_i, P_j \rangle$, denoting a problem configuration, the set of escapable regions, written $\mathcal{R}^e_{(i,j)} \subseteq \text{int}(W)$, is given by $\{R : (R_j, R) \notin \mathit{MVG}\}$. Moreover, for every escapable region $R \in \mathcal{R}^e_{(i,j)}$, there is a set of prevention-from-escape regions, written $\mathcal{R}^p_{(i,j)}(R) \subseteq \text{int}(W)$, given by $\{R' : (R', R) \in \mathit{MVG}\}$.

C. Bound Speed

Given a problem configuration, $\langle E_i, P_j \rangle$, the primary constraint governing pursuit-evasion is given as a relation on two times: the time taken for the evader to reach an escapable region, $t_e(R_{e\langle i,j \rangle})$, for some $R_{e\langle i,j \rangle} \in \mathcal{R}^e_{(i,j)}$, and the time taken for the pursuer to reach one associated prevention-from-escape region, $t_{pe}(R_{pe}(R_{e\langle i,j \rangle}))$, for some $R_{pe}(R_{e\langle i,j \rangle}) \in \mathcal{R}^p_{(i,j)}(R_{e\langle i,j \rangle})$.

For the pursuer to prevent the evader from escaping, the constraint $t_e(R_{e\langle i,j \rangle}) \geq t_{pe}(R_{pe}(R_{e\langle i,j \rangle}))$ must be satisfied at all times, for all $R_{e\langle i,j \rangle} \in \mathcal{R}^e_{(i,j)}$. Considering that both pursuer and evader travel a given path, possibly at a different speed, this constraint can be defined in terms of distances and relative velocities:

$$d_e(P(e), R_{e\langle i,j \rangle}) \geq d_{pe}(P(pe), R_{pe}(R_{e\langle i,j \rangle}))\frac{V_e}{V_{pe}} \quad (1)$$

where V_e and V_{pe} are respectively the speed of the evader and the pursuer and $P(e)$ and $P(pe)$ are the positions of the evader and the pursuer.

This formulation *holds for polygons with or without holes*. However, in polygons with holes a faster evader can always escape pursuer surveillance following a simple strategy: turn around the nearest hole. Conversely, a faster pursuer, without surveillance distance constraint, may apply another simple strategy: catch the evader (moving to a configuration in contact with it) and then stick to it.

However, in polygons without holes, it is possible for a slower pursuer to keep visibility of a faster evader [9].

V. The Effect of Incomplete Information over the Paths to Escape

In general, traveling the shortest-path to reach an escapable region seems to be a good policy for an evader: it is intuitive and easy to realize; moreover, as shown in [9], it is the best policy for the evader if the pursuer knows which escapable region the evader is aiming to. We will show here that this does not hold in the more general case where the pursuer does not know which region among a collection of regions the evader will choose to attempt to escape. In fact, there are cases where an evader can escape provided it does not travel the shortest path to an escapable region. Furthermore, escape would not be possible otherwise. As shown below, this holds regardless of the evader is slower or faster than the pursuer. Let us consider first the case of a faster evader.

Proposition 5.1: There are cases, where a faster evader can escape only it does not travel the shortest distance from its initial position to a escapable region.

Proof: Figure 2 depicts the scenario in which we elaborate our counterexample. There, E stands for the evader, P for pursuer. Let $A(p)$ denote that player A is at distinguished point $p \in \Re^2$.

For the initial system configuration, $(E(2), P(3))$, there are two escapable regions, R_A and R_B, each of which has two prevention from escape regions, $\{R_A, R_{A'}\}$ and $\{R_B, R_{B'}\}$, respectively. Given that strong mutual visibility holds, then if the evader, traveling the shortest path distance, goes to either R_A or R_B, the pursuer is able to prevent escape correspondingly going to either the nearest point that belongs to $R_{A'}$ or $R_{B'}$. $d(E(2), R_A) > d(P(3), R_{A'})\frac{Ve}{Vp}$ and $d(E(2), R_B) > d(P(3), R_{B'})\frac{Ve}{Vp}$. Notice that the pursuer always goes to the nearest prevention from escape region; this explains why going to R_A or R_B is not considered as an option.

Now notice that if the evader first goes to point k, then it will simultaneously diminish the distance to both escapable regions. We emphasize that moving this way the evader is not traveling the shortest path to any of either escapable region (indeed, along this way it is not even moving toward an escapable region). But notice that the pursuer cannot achieve a similar goal: move to a place where the distance to both prevention from escape regions, $R_{A'}$ and $R_{B'}$ simultaneously diminishes. Once at k, the evader has a wining move, given that the evader is faster than the pursuer. This is because $d(E(k), R_A) = d(P(3), R_{A'})$ and $d(E(k), R_B) = d(P(3), R_{B'})$. It follows, that the evader can escape only when it does not travel the shortest path to escape from its initial position. ■

The rationale behind this escape is that the pursuer does not know where the evader is heading at in a long term and so he has to take into account all possible escape regions.

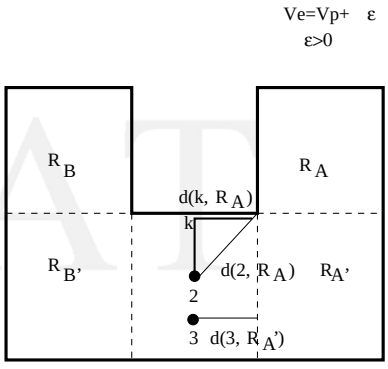

Fig. 2. Evader faster than the pursuer

We now consider the second case, where the evader is slower than the pursuer.

Proposition 5.2: There are scenarios for which the slower evader can escape only if it does not travel the shortest distance from its initial position to a escapable region.

Proof: Refer to Figure 3. Let $A(p)$ denote that player A is at distinguished point $p \in \Re^2$. At first, the evader is at position $E(2)$ and the pursuer at $P(3)$, and thus the system

configuration is $(E(2), P(3))$. Let a and b respectively be the nearest point both to escapable regions R_A and R_B. Notice that in this case these points also belong to $\{R_A, R_{A'}\}$ and $\{R_B, R'_B\}$, the associated prevent from escape regions and they are also the nearest points to prevent escape. Let $L_1 = d(k, a)$, $L_2 = d(2, a)$ and $L_3 = d(3, a)$ and assume both that $L_1 < L_2 < L_3$ and that $\frac{L_1}{L_2} < \frac{L_2}{L_3}$. Without loss of generality, assume that both players move at saturated speed and that $V_p = \frac{L_3}{L_2} V_e$. Then $d(3, 2) > d(2, k)$. Moreover, assume that the time that the evader needs to travel L_2, denoted, $t_e(L_2)$, equals the time the pursuer needs to travel L_3, denoted $t_p(L_3)$.

First, notice that, under these conditions, if the evader attempted to reach a traveling L_2, the pursuer would be able to catch up, traveling L_3. However, if the evader went to k, the pursuer would attempt to move to a place that simultaneously reduces the distance that separates it from both a and b, that is at point 2 (in what follows, we omit from our reasoning the prevention of a escape onto b, but recall that the pursuer must deal with both escape points at once.) But this is impossible. This is because in the new system configuration $(E(k), P(2))$, for the pursuer to catch up it would need to travel at $V_p = \frac{L_2}{L_1} V_e$, but this contradicts our initial assumption, namely: $\frac{L_1}{L_2} < \frac{L_2}{L_3}$. To see this, notice that to catch up in the first step $V_p = \frac{L_3}{L_2} V_e$, but in the second step $V_p = \frac{L_2}{L_1} V_e$ but this implies that the velocity of the pursuer must be bigger than the bound $\frac{L_3}{L_2}$. ■

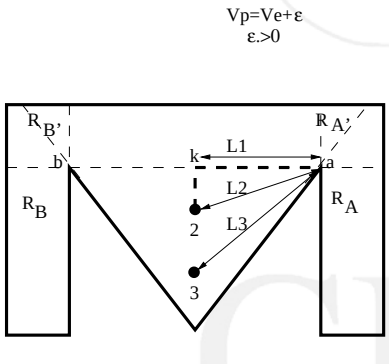

Fig. 3. Evader slower than the pursuer

Thus, together, propositions 5.1 and 5.2, show that there are cases where an evader can escape only if it does not take the shortest path to escape, one of the key contributions of this paper.

In the next sections, we will show that any solution to our problem (a game) depends on two main factors: (i) the initial position of the players; and (ii) the combinatoric paths, together with a map of the environment, that the evader can travel to escape pursuer surveillance in a long-term.

VI. Initial Conditions

The solution to our game (finding a winner) depends on the initial position of both players, as well as their corresponding maximum speed. Clearly, there might be configurations that the pursuer would find unpleasant. Consider, for instance, the case where, even though strong-mutual visibility holds, the players are so apart one another, that, to escape, the evader may just need to go to the adjacent region.

This problem, namely: determining whether the pursuer is able or not to guarantee surveillance for a given evader position, has been investigated before. In [1], for example, the authors proposed a method for the (general) case of classical visibility[1], which works as follows. Given an evader position, the method computes what the authors call a compact set: if the pursuer is outside that set then it will lose the game.

Our approach may perform similarly. For any given analogous configuration, namely the positions of both the pursuer and the evader are given and strong-mutual visibility holds, we can use the MVG and the AG, together with equation (1), to find out whether or not there is a escapable region that the evader can reach in a time strictly smaller than the time needed by the pursuer to reach a corresponding prevent-from-escape region. Notice that d_e and d_{pe} are, in general, *geodesic distances*.

VII. Combinatoric Paths

The solution to our problem *also depends on* fundamental graphs that capture long-term paths taken by the participants. We have found that such graphs can be computed in terms of the map of the environment, considering worst-case scenarios; that is, the best possible paths to escape and the sets of elements in $\Re^2$ such that they prevent the escape even if a best escape path is taken.

In order to compute the sets of elements in $\Re^2$ such that they prevent a escape, we assume that the evader visits reflex vertices traveling along the reduced visibility graph. Notice that a possibility for the evader to escape is to reach a reflex vertex[2]. The reflex vertices break environment convexity and are therefore potential escape points. Any shortest-time path starting on a reflex vertex and visiting any other reflex vertex is a path in the reduced visibility graph [10].

The rationale behind the algorithm below is to find out whether the pursuer can keep surveillance (respectively, the evader can escape) at a long-term, assuming valid initial conditions. The evader travels the reduced visibility graph, choosing a visit ordering which makes the time to escape smaller than the time to prevent escape. Notice that this involves dealing with an intractable problem [9].

Below, we present an algorithm which plans pursuer motions so as to keep track of an evader who does not necessarily travel the shortest paths to an escapable region. This algorithm consists of two methods. The first method uses the network of shortest distance between borders of escapable regions in order to define valid points for the pursuer departure. These points, which depend on the

[1]In classical visibility, two points see one another if the line segment between them does not cross an obstacle at any point other than the endpoints.

[2]A reflex vertex is one of an internal angle greater than π.

velocity of both players, form sets in $\Re^2$, we call Ω *borders*. The second method uses Ω borders in order to identify regions where the pursuer should go to upon each move of the evader. The pursuer knows the motion of the evader only up to a Δt in the future.

It is important to underline that while the Ω borders are computed assuming that the evader travels moving in the network of shortest path between reflex vertices, the Ω borders are used to prevent the escaping of the evader even if it does not move traveling those shortest paths.

We use Ω borders to compute a region in the plane where the pursuer must be in order to prevent the evader from escaping. We call this region S. $S \in \Re^2$ is a set of points, which guarantee that at a given instant of time, the evader cannot reach a reflex vertex in a time strictly smaller than the time that the pursuer needs to reach an Ω border.

A. Ω borders

Algorithm 1 is our method to compute Ω borders.

In general, computing Ω borders requires dealing with a computationally intractable problem. However, we can find an approximate solution, for example, by computing Ω borders for a subset of reflex vertices. This would be useful if we clustered the vertices, hence dividing the tour and then compute Ω borders for each part of this tour. This is equivalent to do local planning; the planning horizon would be determined by the number of vertices to consider. This strategy will not guarantee surveillance at all times but it is useful to make short term planning that prevent evader escaping. Of course, for small polygons, e.g. with around 20 reflex vertices or so, it is actually possible, using a regular PC, to compute the Ω borders for all reflex vertices.

Consider figure 4. In both parts, the environment is the polygon shown with black solid lines, the region partition is shown with dashed lines and the regions are labeled with numbers. The polygon has 4 reflex vertices. The Ω borders, computed setting $Ve = Vp$, are shown in green (light gray) color. In figure 4 A) the Ω borders were computed using the vertices $\{a, b, c\}$; here, the resulting Ω borders are three line segments. In figure 4 B) the Ω borders were computed using all the reflex vertices $\{a, b, c, d\}$; now the resulting Ω borders are simply points. Notice that when the vertex d is considered, the Ω borders are reduced to the vertices themselves. This is because the border of the partition regions and the vertices are equally separated one another. Notice that in this environment, if we set $V_e > V_p$, then the Ω borders would be empty sets. Thus, in this environment, a faster evader will always win.

B. Regions of Local Solution

S is the set of points where the pursuer must be to prevent the evader from going behind a reflex vertex, v_k, and hence escape. Let $\mathbf{V}$ be a subset of all reflex vertices. Then, considering that the evader and the pursuer respectively are at region R_i and R_j, S is defined by (2). There, $\Omega(v_k)$ denotes the Ω border associated to reflex vertex v_k and $\mathbf{V} = \{v \in R : (R, R_j) \notin \textit{MVG}$, that is,

Algorithm 1 Computing Ω borders

Input: Work space, W, and environment partition.
Output: Ω borders.
Remark: In this problem, edge costs are asymmetric (because the borders of the prevent-from-escape regions might be different depending on which way the evader is traveling), *cost*$(v_i, v_j) \neq$ *cost*(v_j, v_i); hence, finding Ω borders requires traversing the tour in one direction (the *clockwise direction*) and then going back in the opposite one (the *counter-clockwise direction*).
1. Find the tour of minimal cost time which visits all the vertexes; call this the *evader tour*;
2. Consider that the evader is at some region which contains one point that touches some vertex (the *initial vertex*) in the tour; call any one such a region a *region associated to the vertex*;
3 Take all the regions associated to the initial vertex; impose a clockwise ordering on these regions; on this ordering, select the first associated region; place the evader in this region. Call the *first escape region* the last region in the previous ordering.
4. Place the pursuer in a region which meets two constraints: (i) the initial constraint, strong mutual visibility of the evader by the pursuer, holds, and (ii) the region is *not* strong mutually visible to the first escape region.
5. Using the evader tour, determine all the escape regions and the corresponding prevent-from-escape regions. This results in two sequences, we call *escape list* and *prevent-from-escape list*, respectively. Notice that for each escape region there still might be more than one prevent-from-escape regions.
for every pair of consecutive regions i and $i+1$ in the escape list **do**
6. Compute the shortest-distance between the borders of regions i and $i+1$, this distance, $d_e(i, i+1)$, is proportional to the time it takes the evader to move between these regions;
7. Decompose regions i and $i+1$; region decomposition returns a set of segments, each of which is a border of the region;
8. For every prevent-from-escape region j associated to region i, decompose region j, call the end result the region-j set;
9. Do likewise for region $i+1$, call the end result the region-$j+1$ set;
10. For every pair of prevent-from-escape regions, one in the set region-j and the other in the set region-$j+1$, collect all the points of each segment of a region in the set region-$j+1$ such that each is at most at a distance $d_e(i, i+1)$ to a segment in a region in the set region-j; we call these segment portion Ω *borders*.
11. return Ω borders
end for

$$S = \{p \in R \quad : \quad (R, R_i) \in MVG) \bigwedge_{v_k \in \mathbf{V}} d(p, \Omega(v_k)) \leq d(P(e), v_k)\frac{V_{pe}}{V_e})\} \tag{2}$$

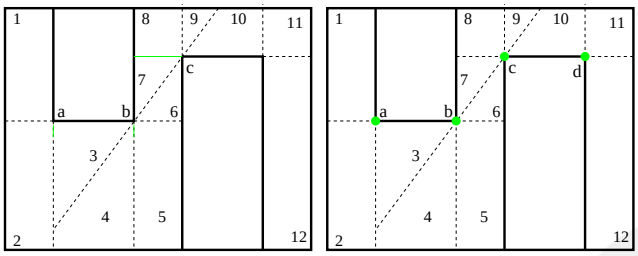

Fig. 4. Ω borders

we consider only the reflex vertices that share a point with some R not strong mutually visible to R_j.

$d(p, \Omega(v_k))$ is the *geodesic distance* between the point p and $\Omega(v_k)$, and $d(P(e), v_k)$ is also the *geodesic distance* between the evader position $P(e)$ and a vertex v_k.

The region S can be used to define a new valid pursuer position regardless the trajectory of the evader. Thus, we have a method to compute the pursuer motion, which is independent of the evader policy and path.

In figure 5, obstacles are shown in gray and the free space in white, regions are delimited with lines. The evader position is shown in red and the Ω borders in green. The associated S region for the pursuer in this scenario is shown in blue.

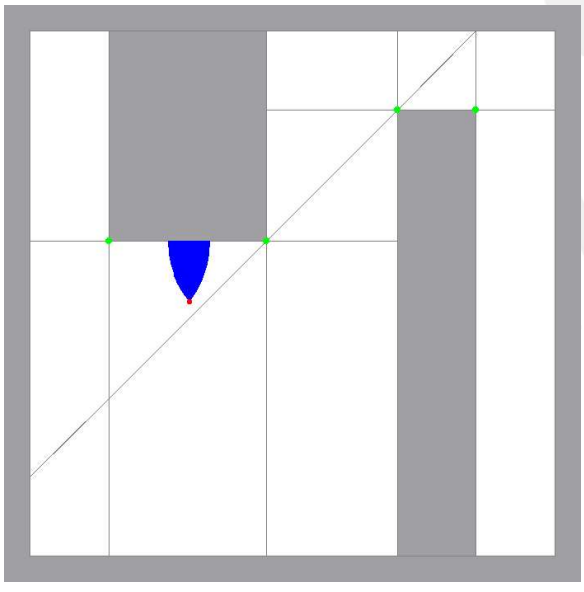

Fig. 5. S region

VIII. Players Strategies and Paths

We have found that under the definition of strong mutual visibility, the possible paths that the evader can travel to escape can be classified in two types: 1) paths where the evader escape when it does not touch a reflex vertice in the environment 2) paths where the evader escape when it touches a vertex in the environment.

The first types of paths does not lie on the visibility graph. Figure 6 shows a path of type 1). As before, the environment is the polygon shown with back solid lines, the region partition is shown with dashed lines and the regions are labeled with numbers. The evader is at region 1 and the pursuer at region 21. If the evader goes to region 2, then the pursuer must go to region 12 (the closest prevention from escape region from the current pursuer position).

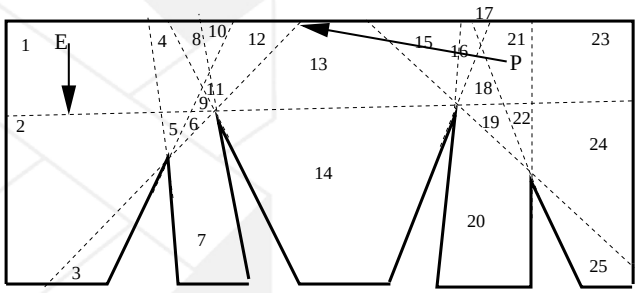

Fig. 6. Paths type 1

The arrows are the evader and pursuer paths. These paths cannot be characterized based only on the reflex vertices positions. But notice that Equation (1) can be used to determine whether or not at a given time moment the evader can escape. At all instants of time, based on the position of the players, together with the MVG and the GV, it is possible to decide whether or not the evader has a winner move.

For the evader travel the second type of paths, he may move along the reduced visibility graph. The motivation for the evader to do so is whenever there is an empty Ω border, it will eventually win (indeed when it reaches the associated vertex). Notice that this condition is independent of the initial position of the players and can be determined using only the map. This is a sufficient condition for the evader to win.

Proposition 8.1: If there is an empty Ω border the evader will eventually win independently of the initial positions of the evader and the pursuer.

Proof: The proof of this condition is immediate by the construction of the Ω borders. ■

IX. Simulation Results

Figure 7 and 8 show snapshots of our simulations. In both figures, obstacles are shown in gray and the free space in white, regions are delimited with line segments. The evader position is the red square and the pursuer is the

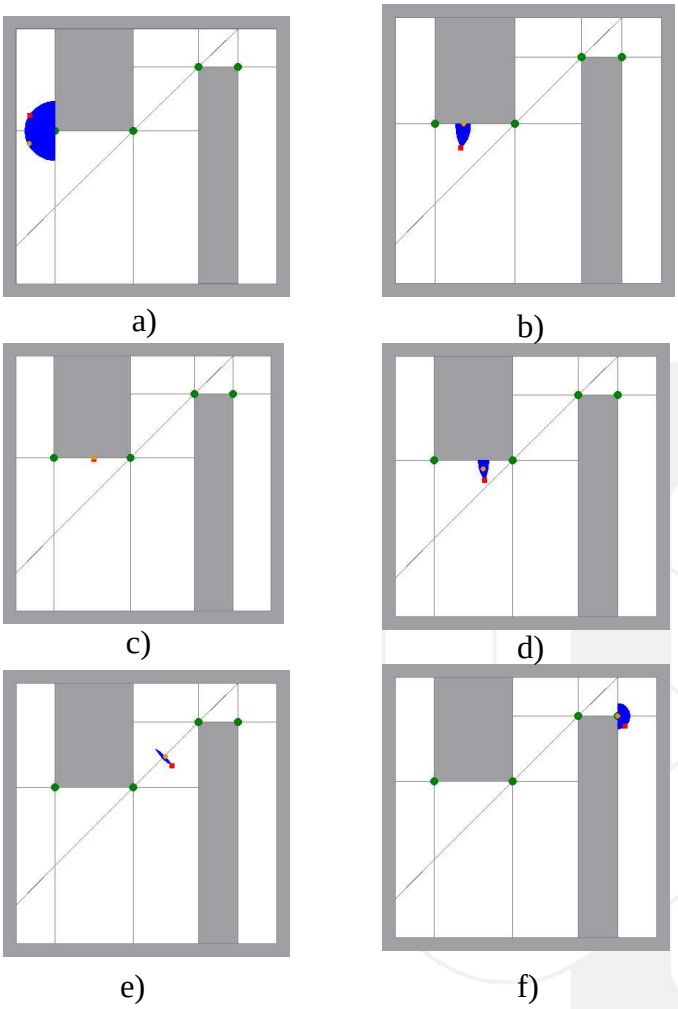

Fig. 7. Simulation Results

orange circle. The Ω borders are shown in green and the S regions in blue.

In both simulations the velocity of the evader was set equal to the velocity of the pursuer $Ve = Vp$. In figure 7, simulation results show that all the Ω borders are points. Here, the evader does not travel the shortest paths to escape. It is interesting to see that when the pursuer gets close to the edges of the reduced visibility graph, the associated S region becomes smaller. In figure 7 c), when the evader touches the obstacle (it is on an edge of the reduced visibility graph), the S region collapses to a single point (the pursuer must be at the same position where the evader is).

Figure 8 shows a bigger environment. In this second simulation, the Ω borders are either points or line segments. Again, the evader does not travel the shortest paths to escape, but whenever it gets close to the edges of the reduced visibility graph, the S region becomes smaller. In fact in figures 8 b) and c), the S regions collapse to a single point.

In both examples, since the environments are small, we compute the Ω borders with an evader tour that consider all

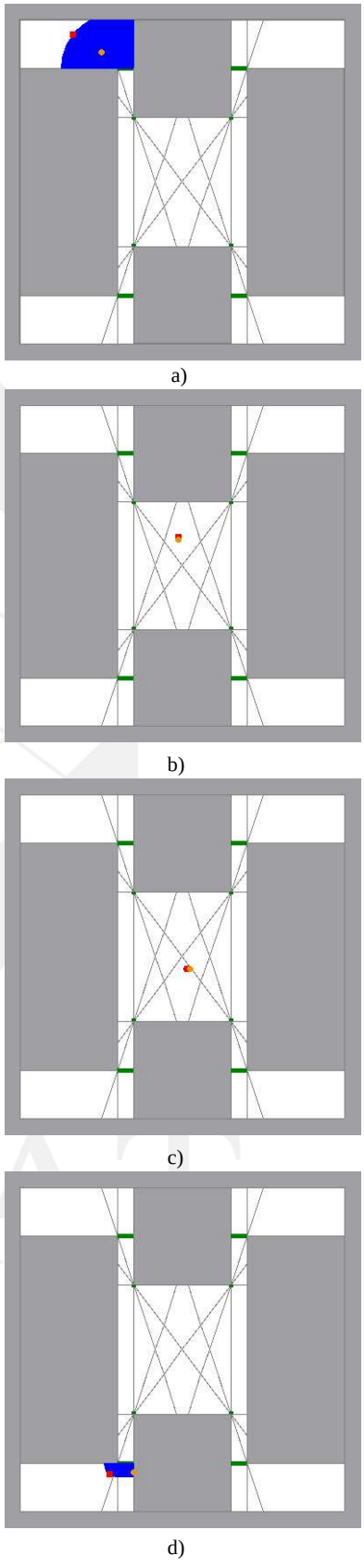

Fig. 8. Simulation Results

the reflex vertices. We are currently testing our programs in significantly bigger environments. In those environments we compute tours that do not visit all reflex vertices. Finally, notice that the S regions are delimited either by line segments or arcs of circles.

X. Conclusion

In this paper we have proved that if the pursuer does not know the evader motion policy then there are cases where an evader can escape only if it does not travel the shortest distance from its initial position to a escapable region, regardless whether the evader is faster or slower than the pursuer. We have presented an algorithm which plans pursuer motions so as to keep track of an evader who does not necessarily travel the shortest paths to an escapable region. We have found a sufficient condition for the evader to escape that does not depend on the initial positions of the players. It only depends on the environment. Therefore, this condition can be checked, before the game starts, if the condition holds then there is no motivation to play. However, notice that finding such a condition implies to solve an NP-complete problem. Finally, we have implemented all our algorithms and presented simulation results.

References

[1] S. Bhattacharya and S. Hutchinson, Approximation Schemes for two-players pursuit evasion games with visibility constraints, *In Proc Int Conf. Robotics Science and Systems IV*, 2008.

[2] K. W. Bowyer and C. R. Dyer, Aspect graphs: An introduction and survey of recent results, *Int. J. Imaging Syst. Technol.*, vol 2, pp. 315-328, 1990.

[3] A. Efrat, H. H. Gonzalez-Baños, S. G. Kobourov, and L. Palaniappan. Optimal Motion Strategies to Track and Capture a Predictable Target. *Proc. IEEE Int. Conf. on Robotics and Automation*, Taipei, Taiwan, pp. 411-423, 2003.

[4] T. Bandyopadhyay, Y. Li, M.H. Ang and D. Hsu, A Greedy Strategy for Tracking a Locally Predictable Target among Obstacles, *In Proc IEEE Int. Conf. on Robotics and Automation*, 2006.

[5] H.H. González, C.-Y. Lee and J.-C. Latombe, Real-Time Combinatorial Tracking of a Evader Moving Unpredictably Among Obstacles, *In Proc IEEE Int. Conf. on Robotics and Automation*, 2002.

[6] O. Tekdas and V. Isler, Robotic Routers. *In Proc IEEE Int. Conf. on Robotics and Automation*, 2008.

[7] B. Jung and G. Sukhatme. Tracking targets using multiple robots: the effect of environment occlusion. In *Journal Autonomous Robots*, vol. 12 pp. 191-205, 2002.

[8] S.M. LaValle, H.H. González-Baños, C. Becker and J.-C. Latombe, Motion Strategies for Maintaining Visibility of a Moving Target. *In Proc IEEE Int. Conf. on Robotics and Automation*, 1997.

[9] R. Murrieta-Cid, R. Monroy, S. Hutchinson and J. P. Laumond A Complexity Result for the Pursuit-Evasion Game of Maintaining Visibility of a Moving Evader, *IEEE International Conference on Robotics and Automation 2008*, pp 2657-2664.

[10] J. O'Rourke, *Computational Geometry In C.* Cambridge University Press, 2000.

[11] L. Parker. Algorithms for Multi-Robot Observation of Multiple Targets. In *Journal Autonomous Robots*, vol. 12 pp. 231-255, 2002.