2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636581

# Roadmap for Visibility-based Target Tracking: Iterative Construction and Motion Strategy

Guillermo Laguna[1], Shashwata Mandal[2], Sourabh Bhattacharya[1,2]
Email: {gjlaguna,shashwata, sbhattac}@iastate.edu

***Abstract*— We consider the problem of generating a fixed path for a mobile observer in a polygonal environment that can maintain a line-of-sight with an unpredictable target. In contrast to purely off-line or on-line techniques, we propose a hierarchical tracking strategy in which an off-line path generation technique based on a RRT is coupled with an on-line feedback-control technique to generate trajectories for the mobile observer.**

## I. Introduction

Motion planning in the presence of uncertainty is a challenging problem [1]. Significant theoretical [2], [3] and technological [4], [5] advances have been made in past three decades to model and control the uncertainty arising from sensor and actuator noise in robotic systems. However, dealing with uncertainty arising from strategic decision-making involving other selfish agents in the environment is still an open area of research [6]. In this work, we address such a scenario which arises in surveillance problems which involve persistently tracking an unpredictable target. Traditional planning considers a joint solution to the problem of minimizing uncertainty and fulfilling the task-level objective. In this work, we consider an alternate approach that decouples the uncertainty (arising from the future actions of the intruder) from the task-level objective (persistent surveillance) for planning the trajectory of the mobile robot. Specifically, we propose a hierarchical tracking strategy in which an off-line sampling-based method [7] is integrated with an on-line feedback-control technique [8] to ensure persistent tracking.

Target-tracking refers to the problem of planning the motion of a mobile observer that tries to track another mobile entity, referred to as the *target*, in an environment containing obstacles. In the past, techniques in differential games [9] have been used to develop feedback-control techniques for guiding the motion of the observer. However, guarantees on persistence of tracking do not scale nicely as the environments become more complex. There have been efforts [10], [11] to provide necessary conditions for persistent tracking based on geometric characteristics of the environment. However, quantifying the observer parameters sufficient for persistent tracking in general environments is still an open problem. For that reason we have decided to address a constrained version of the general problem in which the observer is constrained to move along a fixed path.

Department of Mechanical Engineering[1], Department of Computer Science[2], Iowa State University, Ames, IA-50011.

This work was in part supported by NSF IIS award 1816343.

A necessary condition for persistent tracking is coverage. A closed path which covers the entire region when a mobile observer is constrained to move along it called a *watchman's route* [12], [13], [14]. The problem of computing minimum length watchman's route is referred to as the *watchman's route problem*. The watchman's route problem can be solved in polynomial time when the region to be guarded is a simple polygon but it is NP-hard for polygons with holes [15]. In [16], we use properties of a watchman route in closed polygonal environments to construct a fixed route for a mobile observer to persistently track an unpredictable target (called the *paparazzi route* or $p$-route). Additionally, we prove the existence of a tree on which the observer can minimize its tracking speed. This motivates our current work which explores sampling-based techniques, specifically Rapidly exploring Random Trees (RRT) in constructing a $p$-route. The advantage of using RRTs is threefold; (1) The basic RRT can be modified to rapidly generate a tree that can cover an environment (2) Sampling based planners can generate paths in high-dimensional configuration spaces [17], [18] (3) Kinodynamic planners can handle systems with dynamics [19]. Although, the focus of our current work is (1), (2) and (3) render the proposed technique appealing for extensions in future.

The main contributions of this work are as follows: (i) Given a closed polygonal environment, we propose a sampling-based strategy to build a $p$-route. (ii) Given the initial $p$-route, we propose a strategy to modify it in order to reduce the speed required by the observer for tracking. This provides a specification of the observer parameters for the tracking task in real scenarios. To summarize, the overall contribution of this work lies in building a bridge between sampling-based techniques [19] and art-gallery problems [20] to address problems in vision-based pursuit.

The paper is organized as follows. In Section II, we present the problem formulation. In Section III, we present a sampling-based technique to construct the $p$-route. In Section V, we present the motion strategy for the observer on the $p$-route. In Section VI, we define a metric to find a non-trivial upper bound for the minimum speed required by the observer to guarantee persistent tracking. In Section VII, we propose a method to simplify the path obtained in Section III. Finally, we present the conclusions and future research directions in Section VIII.

## II. Problem Formulation

Consider a closed environment represented as a simple polygon $P$. An intruder moves inside $P$ with bounded speed. The intruder is assumed to be unpredictable, i.e., there is no prior knowledge about its future actions. A mobile observer, referred to as the *guard*, tries to maintain a line-of-sight (LOS) with the intruder as it moves around in $P$. In this work, we assume that the guard has an omni-directional field-of-view with infinite range. Let $x_I = x_I(t) \in P$ and $0 \leq v_I(t) \leq \overline{v}_I < \infty$ denote the instantaneous location and speed of the intruder at time $t$, respectively, where $\overline{v}_I$ denotes the maximum speed of the intruder. The instantaneous location and speed of the guard at time $t$ are denoted by $x_g = x_g(t) \in P$ and $0 \leq v_g(t) \leq \overline{v}_g < \infty$ respectively, where $\overline{v}_g$ is the maximum speed of $g$. Additionally, we assume that guard is constrained to move along a pre-specified path inside $P$, called a $p$-route which is computed off-line. Given $x_I(0)$ and $\overline{v}_I$, we investigate the problem of finding a $p$-route $\gamma$, the initial location of the guard $x_g(0) \in \gamma$ and a motion strategy for the guard using techniques from sampling-based methods.

## III. Construction of a $P-$Route

### A. Relevant Concepts and Results from Previous Work [16]

In this section, we mention the relevant results from [16] regarding the design of a $p-$route and introduce terminology used in the rest of the analysis.

Let $V(P)$ be the vertex set of $P$ and $V^{rf}(P) \subset V(P)$ the subset of reflex vertices. A star region (of a reflex vertex) $R(v_i) \subset P$, where $v_i \in V^{rf}(P)$, is the set of points in $P$ visible from $v_i$ that lie inside the region obtained by extending the two edges of $E(P)$ (edge set of $P$) that have $v_i$ as an endpoint. The extended edges are called *cuts* [14]. Any route that visits a specific subset of cuts of a given environment, called *essential cuts*, is a watchman's route [14]. Thus, any route that visits all the essential cuts is a candidate $p-$route [16]. In Figure 1 (a), an environment with two reflex vertices $v_i$ and $v_j$ is shown along with their star region $R(v_i)$ and $R(v_j)$, respectively.

In [16], it is shown that $\gamma$ ($p$-route) consists of **a set of line segments**. As an abstraction, a connected graph $G = G(\gamma)$ can be used to represent $\gamma$, where each line segment of $\gamma$ corresponds to an edge in $E(G)$ (edge set of $G$), and the endpoints of those segments correspond to the vertices in $V(G)$ (vertex set of $G$). Additionally, it is shown in [16] that there exists a tree on which the guard can achieve the minimum speed for persistent tracking. Therefore, we consider the case in which $G$ is a tree. The following terminology is introduced: $s_{i,\gamma} = R(v_i) \cap \gamma$ is the set of points in $\gamma$ inside $R(v_i)$, $S_{V(P),\gamma}$ is the collection of all $s_{i,\gamma}$ sets, where each $i$ corresponds to $v_i \in V^{rf}(P)$. Moreover, $S_{p,\gamma} \subset S_{V(P),\gamma}$ is a set of *representative points*, i.e., for each $p_i \in S_{p,\gamma}$, there is a $s_{i,\gamma} \in S_{V(P),\gamma}$, such that $p_i \in s_{i,\gamma}$ and $|S_{p,\gamma} \cap s_{i,\gamma}| = 1$, which means that it is a collection of points in $\gamma$ such that each one of them is inside a different region $R(v_i)$.

There exists an optimal set of representative points $S^*_{p,\gamma}$, such that the minimum speed required to guarantee persistent tracking $v_{i^*,j^*}$ when $g$ moves along $\gamma$ is equal to the minimum speed required to guarantee persistent tracking when $g$ is forced to reach a representative point $p_i \in S^*_{p,\gamma}$ to prevent $I$ from breaking the LOS when it approaches $v_i$. Thus, building a $p-$route is equivalent to construct a tree in $P$ such that it has one vertex for each point in $S_{p,\gamma}$ [16]. Such a minimum speed is denoted by $v_{i^{max},j^{max}}(S^*_{p,\gamma}) = v_{i^*,j^*} = \overline{v}_I \max\{z(\gamma_{i,j}) : (i,j) \in \mathbb{Z}_\gamma\}$, where $\gamma_{i,j}$ is the path between $p_i$ and $p_j$, $\mathbb{Z}_\gamma = \{(i,j) \in \mathbb{Z}^2 : \gamma_{i,j} \subseteq \gamma \text{ and } p_i, p_j \in S^*_{p,\gamma}\}$ is the set of all pairs of indexes of reflex vertices and $z(\gamma_{i,j}) = \frac{d(p_i,p_j)}{d(R_{i,j}(p_i),R_{i,j}(p_j))}$, is the "cost" associated to $\gamma_{i,j}$. $d(R_{i,j}(p_i), R_{i,j}(p_j))$ is the distance between the regions $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$, which are defined as the regions such that when $x_g \in \gamma_{i,j}$, $x_I \in R_{i,j}(p_i)$ (or $x_I \in R_{i,j}(p_j)$), the LOS between $I$ and $g$ is lost, unless $x_g = p_i$ with $p_i \in \gamma \cap R(v_i)$ (or $x_g = p_j$ with $p_j \in \gamma \cap R(v_j)$, analogously).

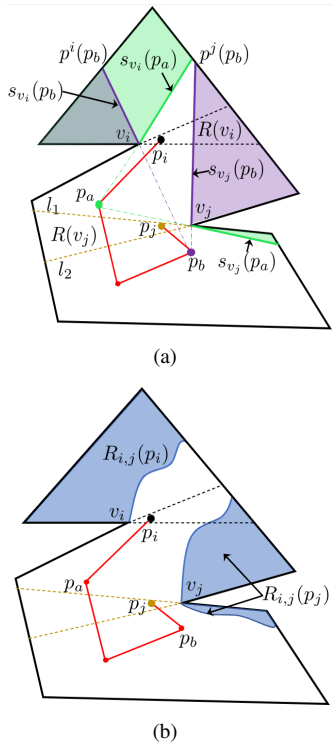



Fig. 1: (a) $s_{v_i}$ and $s_{v_j}$ are obtained from different points along the path, (b) Representation of the $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$ regions.

In Figure 1(a), there is a polygon with two reflex vertices. The red shaded path is a valid $p-$route since it is an open path that visits the two star regions. Assume that $S_{p,\gamma} = \{p_i, p_j\}$. Thus, $x_g = p_i$ to prevent $I$ from using $v_i$ to break the LOS (analogously, $x_g = p_j$ prevents $I$ from using $v_j$ to break the LOS). From [16], for any $p_a \in \gamma_{i,j}$ and a reflex vertex $v_i$, $s_g^{v_i}$ is defined as the longest line segment in $P$ that passes through $v_i \in s_g^{v_i}$ and which has $p_a$ and $p^i(p_a) \in \delta P \backslash \{v_i\}$ as its endpoints.

Thus, $s_{v_i}(p_a) \subset s_g^{v_i}$ is the directed segment from $v_i$ to $p^i(p_a)$. In Figure 1(a), $s_{v_i}(p_a)$ and $s_{v_j}(p_a)$ are illustrated as green shaded segments. Moreover, $s_{v_i}(p_b)$ and $s_{v_j}(p_b)$ are illustrated as purple segments. Notice that if $x_g = p_a$, as long as $I$ is outside the green and gray shaded regions, the LOS between $I$ and $g$ is not broken (analogously, if $x_g = p_b$, as long as $I$ is outside the purple and gray shaded regions the LOS between $I$ and $g$ is not broken). Since the orientation, and consequently, the distance between $s_{v_i}$ and $s_{v_j}$ depend on the location along $\gamma_{i,j}$ [16], $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$ are usually unknown. When $V^{rf}(P) = \{v_i, v_j\}$ and $d(x_I, R_{i,j}(p_i)) = \min\{d(x_I, q_i) : q_i \in R_{i,j}(p_i)\}$, $\overline{v}_g \geq \overline{v}_I \frac{d(x_g,p_i)}{d(x_I,R_{i,j}(p_i))}$ and $\overline{v}_g \geq \overline{v}_I \frac{d(x_g,p_j)}{d(x_I,R_{i,j}(p_j))}$ are sufficient conditions to guarantee persistent tracking. Hence, tracking may be lost if $x_g \in \gamma_{i,j} \subset \gamma$, and it needs to move towards $p_i \in R(v_i) \cap \gamma$ and $p_j \in R(v_j) \cap \gamma$ at the same time, which happens when,

$$\overline{v}_g < \overline{v}_I \frac{d(p_i, p_j)}{d(R_{i,j}(p_i), R_{i,j}(p_j))}, \tag{1}$$

where $d(R_{i,j}(p_i), R_{i,j}(p_j)) = \min\{d(q_i, q_j) : q_i \in R_{i,j}(p_i) \text{ and } q_j \in R_{i,j}(p_j)\}$, $d(q_i, q_j)$ is the length of the shortest path inside $P$ between $q_i$ and $q_j$, and $d(p_i, p_j)$ is the distance along $\gamma$ between $p_i$ and $p_j$. Consequently, $\overline{v}_g \geq \overline{v}_I \frac{d(p_i,p_j)}{d(R_{i,j}(p_i),R_{i,j}(p_j))}$, guarantees that $g$ is able to persistently track $I$. Although we do not know the exact definition of such $R_{i,j}(p_i)$ regions, it suffices to assume that they are given to provide guarantees for the minimum speed required by $g$ when following the motion strategy from Section V. In Figure 1 (b) the $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$ regions that correspond to $v_i$ and $v_j$ are illustrated.

## IV. A Sampling Based $P-$Route

Based on the discussion in Section III-A, we propose a technique to construct a $p-$route which (we believe) is an improvement over the approach presented in [16]. Since the construction of the $p-$route has been reduced to the problem of constructing a tree that visits all the essential cuts of $P$, we use a sampling-based approach to construct $\gamma$. Specifically, we use a variation of the standard Rapidly Exploring Random Tree (RRT) [21] to construct the tree. Algorithm 1 provides a pseudo-code of our technique. It starts from an arbitrary point along an essential cut, and ends when the tree has reached all the remaining essential cuts. Moreover, a vertex of the tree is assigned to each star region which guarantees that we have a set of representative points visited along the traversal of the tree. Figure 2 (a) illustrates the output generated by Algorithm 1. The red segments represent the cuts in the set of essential cuts, and the gray dashed lines represent the boundaries of the star regions.

Let $S_c$ be the set of essential cuts, $B(P)$ be a two-dimensional bounding box containing $P$, $c_r > 0$ be an arbitrary maximum distance between two points in the path generated by the RRT and $Steer(x_{nearest}, x_{rand})$ be a function that returns a point along the segment $s_{near}$ (defined in line 13 of Algorithm 1) whose distance to $x_{nearest}$ (defined in line 11 of Algorithm 1) is less than $c_r$,

$$Steer(x_{nearest}, x_{rand}) = \begin{cases} x_{nearest} + \frac{s_{near}}{\|s_{near}\|}, & \|s_{near}\| < c_r \\ x_{rand}, & \text{otherwise} \end{cases}, \tag{2}$$

where $\|\cdot\|$ stands for the Euclidean norm in $\mathbb{R}^2$.

We are interested in the tree $G_{prun} \subseteq G$ that contains the paths $\gamma_{i,j}$ between each pair of representative points in $S_{p,\gamma}$. Thus, $S_{p,\gamma} \subseteq V(G_{prun})$ and every leaf in $G_{prun}$ corresponds to a point in $S_{p,\gamma}$ (see Figure 2 (b)). Algorithm 2 details the procedure to generate $G_{prun}$. It takes $O(n)$ time, where $n = |V(G)|$ is the cardinality of $V(G)$. Some of the vertices that are not in $S_{p,\gamma}$ correspond to points in $\gamma$ where different branches of the tree intersect while others are necessary to avoid trajectories that lead to collision with the boundary of $P$.

**Algorithm 1** Initial RRT

1: **Input**: $P$, $S_c$, $c_r$
2: **Output**: $\gamma$, $G$, $S_{p,\gamma}$
3: $E(G) \leftarrow \emptyset$, $\overline{S}_c \leftarrow S_c$
4: $s_{init} \leftarrow$ randomly chosen segment from $S_c$
5: $x_{init} \leftarrow$ randomly selected location along $s_{init}$
6: $\overline{S}_c \leftarrow \overline{S}_c \backslash \{s_{init}\}$, $V(G) \leftarrow \{v(x_{init})\}$
7: update $\gamma$
8: $S_{p,\gamma} \leftarrow \{x_{init}\}$
9: **while** $\overline{S}_c \neq \emptyset$ **do**
10: $x_{rand} \leftarrow$ random sample in $B(P)$
11: $x_{nearest} \leftarrow$ point in $V(G)$ closest to $x_{rand}$
12: $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$
13: $s_{near} \leftarrow x_{rand} - x_{nearest}$
14: **if** $s_{near} \subset P$ **then**
15: **if** $s_{near}$ intersects a segment $s_{ess} \in \overline{S}_c$ **then**
16: $x_{new} \leftarrow s_{near} \cap s_{ess}$
17: $S_{p,\gamma} \leftarrow S_{p,\gamma} \cup \{x_{new}\}$
18: $\overline{S}_c \leftarrow \overline{S}_c \backslash \{s_{ess}\}$
19: **end if**
20: $V(G) \leftarrow V(G) \cup \{v(x_{new})\}$, $E(G) \leftarrow E(G) \cup \{(v(x_{nearest}), v(x_{new}))\}$
21: **end if**
22: **end while**
23: return $\gamma$,$G$,$S_{p,\gamma}$

## V. Tracking Strategy

In this section, we present present algorithms to determine the initial position of the guard, and its subsequent tracking strategy. Let $\gamma$, $S_{p,\gamma}$ and $x_I(0) \in P$ be given. We define $S_\gamma$ as the set of all paths $\gamma_{i,j} = \gamma_{j,i}$ such that $p_i, p_j \in S_{p,\gamma}$ and $i \neq j$. From (1), the minimum speed that guarantees persistent tracking when the intruder follows the shortest path between $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$ ($x_g = p_i$ when $x_I \in R_{i,j}(p_i)$ and $x_g = p_j$ when $x_I \in R_{i,j}(p_j)$) is $v_{i,j} = \overline{v}_I \frac{d(p_i,p_j)}{d(R_{i,j}(p_i),R_{i,j}(p_j))}$. Let $v_{i^*,j^*} = \max_{p_i,p_j \in S_{p,\gamma}} \{v_{i,j}\}$.

Algorithm 3 finds an initial location of the guard ($x_g(0)$) visible to the initial location of the intruder ($x_I(0)$). The

**Algorithm 2** Pruning Strategy

**Input**: $G$
**Output**: $G_{prun}$, $\gamma$
$G_{prun} \leftarrow G$
$V_{leaves}(G_{prun}) \leftarrow$ leaves of $G_{prun}$
**while** $V_{leaves}(G_{prun}) \cap (V(G_{prun}) \backslash S_{p,\gamma}) \neq \emptyset$ **do**
    $v \leftarrow$ vertex in $V_{leaves}(G_{prun}) \cap (V(G_{prun}) \backslash S_{p,\gamma})$
    $V(G_{prun}) \leftarrow V(G_{prun}) \backslash \{v\}$
    update $E(G_{prun})$,$V_{leaves}(G_{prun})$
**end while**
update $\gamma$
return $G_{prun}$,$\gamma$

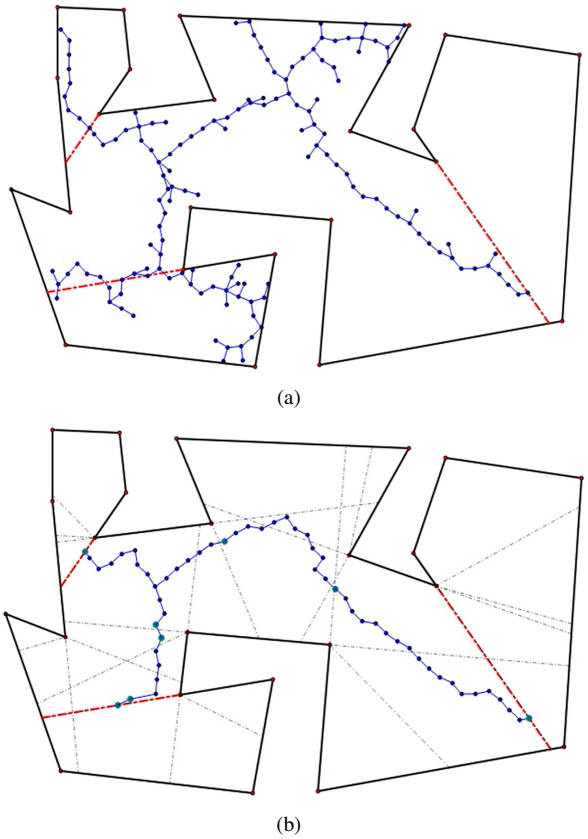
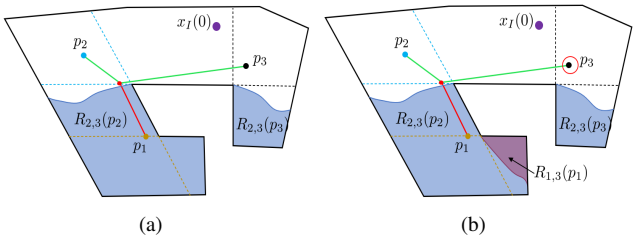


Fig. 2: (a) Initial path obtained through a Algorithm 1. (b) Path obtained after Algorithm 2.

number of representative points is $O(n)$ ($n$ be the number of vertices of $P$) and the number of $\gamma_{i,j}$ paths is $O(n^2)$. Consequently, Algorithm 3 takes $O(n^3)$ time. Figure 3(a) shows an environment with three reflex vertices, the initial location of the intruder, $\gamma$ and its representative points $p_1$, $p_2$ and $p_3$. $\gamma_{2,3}$ is shown in green along with the $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$ regions. Figure 3(b) illustrates the case where $p_1$ is also considered, its corresponding region $R_{1,3}(p_1)$ is shown since the figure represents the step to determine $S(p_1) = \{p \in \gamma_{1,3} : d(p,p_1) \leq d_{1,3}(p_1)\}$, where $d_{1,3}(p_1) = \frac{v_{i^*,j^*}}{\overline{v}_I} d(R_{1,3}(p_1), x_I(0))$. Algorithm 3 finds a location of the guard on $\gamma$ from where the intruder is visible initially. In Figure 4(a), the subpath enclosed by the red circle represents a location where the aforementioned inequalities are satisfied, and $x_I(0)$ is visible from $x_g(0)$. However, there may be cases in which no initial position of the guard exists from where the intruder is visible. Figure 4(b) illustrates a case where not all the inequalities are satisfied ($d_{1,3}(p_3)$ is shorter) and consequently $S(x_g(0)) = \emptyset$. Figure 4(c) illustrates another case in which the path enclosed by the red circle represents a location where the inequalities are satisfied but $x_I(0)$ is not visible is shown. Therefore, both the visibility constraints and reachability constraints (to $R_{i,j}$(p˙i) and $R_{i,j}$(p˙j)) need to be satisfied at the initial location of the guard.




(a) (b)

Fig. 3: (a) Path $\gamma_{2,3}$ and its regions $R_{2,3}(p_2)$ and $R_{2,3}(p_3)$, (b) $p_1$ is considered.

**Algorithm 3** Computation of $x_g(0)$

**Input**: $S_\gamma$,$\gamma$,$x_I(0)$,$P$
**Output**: $x_g(0)$
**for** each path $\gamma_{i,j} \in S_\gamma$ **do**
    $d_{i,j}(p_i) \leftarrow \frac{v_{i^*,j^*}}{\overline{v}_I} d(R_{i,j}(p_i), x_I(0))$
    $d_{i,j}(p_j) \leftarrow \frac{v_{i^*,j^*}}{\overline{v}_I} d(R_{i,j}(p_j), x_I(0))$
    $S(x_g(0)) \leftarrow \{p \in \gamma_{i,j} : d(p,p_i) \leq d_{i,j}(p_i)$ and $d(p,p_j) \leq d_{i,j}(p_j)\}$
    $S'_{i,j} \leftarrow S'_{p,\gamma} \backslash \{p_i, p_j\}$
    **for** each $p_k \leftarrow\in S'_{i,j}$ **do**
        $d_{i,k}(p_k) \leftarrow \frac{v_{i^*,j^*}}{\overline{v}_I} d(R_{i,k}(p_k), x_I(0))$
        $S(p_k) \leftarrow \{p \in \gamma_{i,k} : d(p,p_k) \leq d_{i,k}(p_k)\}$
        $d_{j,k}(p_k) \leftarrow \frac{v_{i^*,j^*}}{\overline{v}_I} d(R_{j,k}(p_k), x_I(0))$
        $S(p_k) \leftarrow S(p_k) \cap \{p \in \gamma_{j,k} : d(p,p_k) \leq d_{j,k}(p_k)\}$
        $S(x_g(0)) \leftarrow S(x_g(0)) \cap S(p_k)$
    **end for**
    **if** $S(x_g(0)) \neq \emptyset$ **then**
        $x_g(0) \leftarrow$ arbitrary location in $S(x_g(0))$
        Go to 20
    **end if**
**end for**
return $x_g(0)$

After a feasible $x_g(0)$ is obtained from Algorithm 3, tracking requires an update in the location of the guard for $t > 0$ depending on the motion of the intruder. Algorithm 4 is a polynomial time algorithm to compute of the location

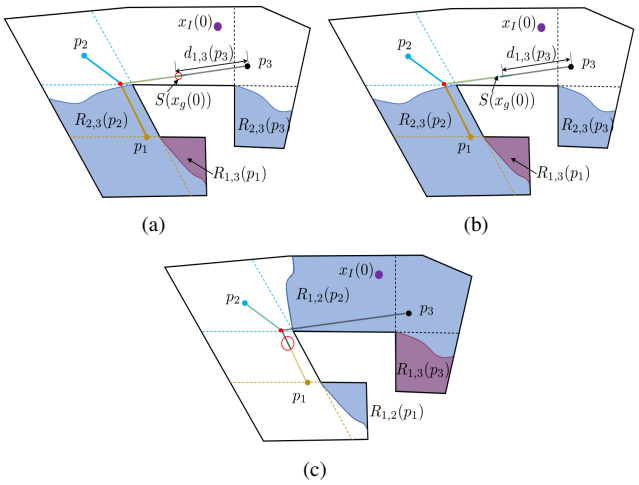



Fig. 4: Example of: (a) A region $S(x_g(0)) \neq \emptyset$, (b) $d_{1,3}(p_3)$ is shorter so $S(x_g(0)) = \emptyset$, (c) Impossible case where $S(x_g(0)) \neq \emptyset$ and $x_I(0)$ is not visible from $S(x_g(0))$.

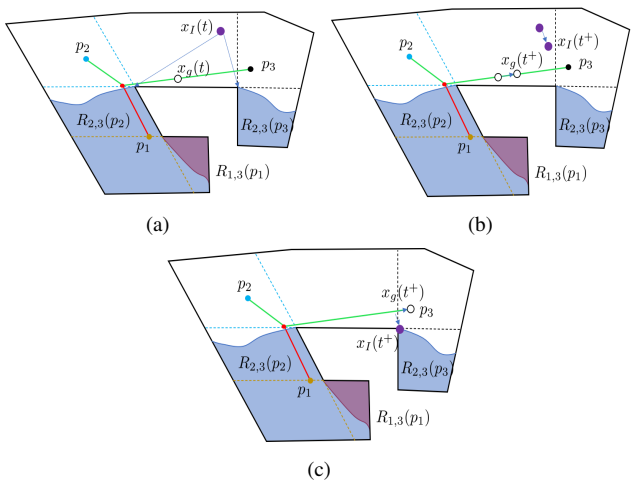



Fig. 5: (a) Location of intruder and guard at time $t$, (b) $x_g(t^+)$ when $I$ approaches $R_{2,3}(p_3)$, (c) $x_g(t^+) = p_3$.

**Algorithm 4** Update of $x_g(t)$

**Input**: $x_I(t^+)$,$S_\gamma(x_g(t))$,$S_{p,\gamma}$
**Output**: $x_g(t^+)$
$S_\gamma^{path} \leftarrow \emptyset$
**for** each $p_i \in S_{p,\gamma}$ **do**
  $S_\gamma^{cand} \leftarrow \emptyset$
  **for** each $\gamma_{i,j} \in S_\gamma(x_g(t))$ **do**
    **if** $d(x_I(t), R_{i,j}(p_i)) > d(x_I(t^+), R_{i,j}(p_i))$ **then**
      $S_\gamma^{cand} \leftarrow S_\gamma^{cand} \cup \{\gamma_{i,j}\}$
    **end if**
  **end for**
  $\gamma_{cand} \leftarrow \arg\min_{\gamma_{i,j} \in S_\gamma^{cand}} \{d(x_I(t^+), R_{i,j}(p_i))\}$
  $S_\gamma^{path} \leftarrow S_\gamma^{path} \cup \{\gamma_{cand}\}$
**end for**
$\gamma_{i^{max},j^{max}} \leftarrow \arg\max_{\gamma_{i,j} \in S_\gamma^{path}} \{d(x_I(t), R_{i,j}(p_i)) - d(x_I(t^+), R_{i,j}(p_i))\}$
$x_g(t^+) \leftarrow$ location along $\gamma_{i^{max},j^{max}}$ such that $d(x_g(t^+), p_{i^{max}}) = \frac{v_{i^*,j^*}}{\overline{v}_I} d(x_I(t^+), R_{i^{max},j^{max}}(p_{i^{max}}))$
return $x_g(t^+)$

of guard at the next instant, $x_g(t^+)$. Figure 5(a) shows a guard located at $x_g(t)$ when the intruder is located at $x_I(t)$. From that location, the intruder can try to reach $R_{2,3}(p_2)$ or $R_{2,3}(p_3)$. In Figure 5(b), the intruder moves towards $R_{2,3}(p_3)$ so the guard reacts accordingly by following Algorithm 4, which guarantees that when $x_I(t^+) \in R_{2,3}(p_3)$, $x_g(t^+) = p_3$ (refer to Figure 5(c)).

## VI. Approximation of $R_{i,j}(p_i)$

In the previous sections, we assumed the regions $R_{i,j}(p_i)$ for each $\gamma_{i,j} \in S_\gamma$ were known and that such regions were "the optimal" ones, in the sense that the speeds $v_{i,j} = \overline{v}_I \frac{d(p_i,p_j)}{d(R_{i,j}(p_i),R_{i,j}(p_j))}$ were the minimum speeds that guarantee persistent tracking when the intruder moves between $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$ while the guard moves between $p_i$ and $p_j$. Let $s_g^{v_i} = s_g^{v_i}(x_g(t))$ be the longest line segment lying entirely in $P$ such that $v_i \in s_g^{v_i}$ and $x_g(t)$ is an endpoint of $s_g^{v_i}$. We define $p^i(x_g) \in \delta P \backslash \{v_i\}$ as the opposite endpoint of $s_g^{v_i}$. Next, we define $s_{v_i} = s_{v_i}(x_g(t)) \subset s_g^{v_i}$ as the directed segment from $v_i$ to $p^i(x_g)$. As long as the intruder lies to the left of (or at) $s_{v_i}$, the LOS between the intruder and the guard is not broken by $v_i$. Visibility is lost as soon as the intruder lies to the right of $s_{v_i}$. Preventing the intruder from breaking the LOS is equivalent to prevent it from reaching the right side of any $s_{v_i}$. Let $A_{v_i}(x_g(t)) \subset P$ denote the region located to the right of $s_{v_i}(x_g(t))$. Since $s_{v_i}(x_g(t))$ and $A_{v_i}(x_g(t))$ depend on $x_g(t)$, we define $R'_{i,j}(p_i) = \bigcup_{p_k \in \gamma_{i,j}} A_{v_i}(x_g(t))$, and $R'_{i,j}(p_j) = \bigcup_{p_k \in \gamma_{i,j}} A_{v_j}(x_g(t))$ as approximations of $R_{i,j}(p_i)$ and $R_{i,j}(p_j)$, respectively. Regardless of the location of the guard along $\gamma_{i,j}$, the intruder cannot use $v_i$ to break the LOS as long as $x_I(t) \notin R_{i,j}(p_j)$. Therefore, these are valid approximations. However, since they are approximations, $d(R'_{i,j}(p_i), R'_{i,j}(p_j)) \leq d(R_{i,j}(p_i), R_{i,j}(p_j))$. Hence we define $v'_{i^*,j^*} = \overline{v}_I \frac{d(p_i,p_j)}{d(R'_{i,j}(p_i),R'_{i,j}(p_j))}$ instead of $v_{i^*,j^*}$. Clearly, $v'_{i^*,j^*} \geq v_{i^*,j^*}$.

## VII. Simplification of $\gamma$

In this section, we present a procedure to simplify $G_{prun}$ returned by Algorithm 2. For every path $\gamma_{i,j}$ in $G_{prun}$, we use $R'_{i,j}(p_i)$ and $R'_{i,j}(p_j)$ as defined in Section VI. Algorithm 5 details such a procedure. It visits every vertex in $V(G_{prun})$ that does not correspond to a representative point in $S_{p,\gamma}$, so the algorithm takes linear time. Since all the leaves in $V(G_{prun})$ correspond to representative points, every vertex in $V(G_{prun}) \backslash S_{p,\gamma}$ has at least degree 2. Let $deg(v(p_i))$ denote the degree of vertex $v(p_i)$. If $deg(v(p_i)) > 2$, the point $p_i \in \gamma$ is an intersection point in the path $\gamma$ and therefore, not deleted from $G_{sim}$. If $deg(v(p_i)) = 2$, $v(p_j)$ and $v(p_k)$ are the neighbors of $v(p_i)$. By definition, $\gamma_{j,k}$ consists of two connected line segments

$\gamma_{j,i}$ and $\gamma_{i,k}$. Since $p_i$ is not a representative point, $\gamma_{j,k}$ is simplified by replacing it with a line segment between $p_j$ and $p_k$. Algorithm 5 checks if such a line segment lies inside $P$, and if that is the case, the path and its corresponding graph $G_{sim}$ are modified. Figure 6 illustrates an approximate *p-route* obtained from Algorithm 5 in a polygonal environment.

**Algorithm 5** Simplify Strategy

**Input**: $G_{prun}$, $P$,$S_\gamma$
**Output**: $G_{sim}$, $\gamma$,$S_\gamma$
$G_{sim} \leftarrow G_{prun}$
**for** each $v(p_i) \in V(G_{sim}) \backslash S_{p,\gamma}$ **do**
  **if** $deg(v_i) = 2$ **then**
    $v(p_j) \leftarrow$ first neighboring vertex of $v_i$
    $v(p_k) \leftarrow$ second neighboring vertex of $v_i$
    **if** $\gamma_{j,k} \in P$ **then**
      $V(G_{sim}) \leftarrow V(G_{sim}) \backslash \{v(p_i)\}, S_\gamma \leftarrow S_\gamma \backslash \{p_i\}$
      $E(G_{sim}) \leftarrow E(G_{sim}) \backslash \{(v(p_j), v(p_i)), (v(p_i), v(p_k))\}$
      $E(G_{sim}) \leftarrow E(G_{sim}) \cup \{(v(p_j), v(p_k))\}$
      update $\gamma$
    **end if**
  **end if**
**end for**
return $G_{sim}$,$\gamma$,$S_\gamma$

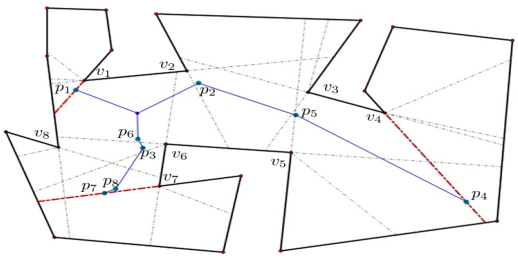



Fig. 6: Path obtained after using Algorithm 5.

## VIII. Conclusion

In this work, we considered the problem of generating path for a mobile observer in a polygonal environment that can maintain a line-of-sight with an unpredictable target. In contrast to purely off-line or on-line techniques, we proposed a hybrid tracking strategy in which an off-line path generation technique based on a sampling-based method is coupled with an on-line feedback-control technique to ensure persistent tracking. Finally, we proved that the proposed tracking strategy leads to a constant-factor of approximation for the observer's speed from its optimal. As part of our future research, we plan to extend our current technique to observers having higher-order dynamics which will lead to planning in high-dimensional configuration spaces. Additionally, we plan to explore the challenges in implementation as well as evaluate the efficacy of our proposed technique in practical settings with real robots. For that, we need to tackle the problem of uncertainty in sensing and actuation in addition to lack of information about the future actions of the target.

## References


[1] J.-C. Latombe, A. Lazanas, and S. Shekhar, "Robot motion planning with uncertainty in control and sensing," *Artificial Intelligence*, vol. 52, no. 1, pp. 1 – 47, 1991.

[2] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 723–730.

[3] J. Van den Berg, P. Abbeel, and K. Goldberg, "Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[4] J. Chu, K. Zhao, Q. Zhang, and T. Wang, "Construction and performance test of a novel polarization sensor for navigation," *Sensors and Actuators A: Physical*, vol. 148, no. 1, pp. 75 – 82, 2008.

[5] S. A. Hiremath, G. W. [van der Heijden], F. K. [van Evert], A. Stein, and C. J. [ter Braak], "Laser range finder model for autonomous navigation of a robot in a maize field using a particle filter," *Computers and Electronics in Agriculture*, vol. 100, pp. 41 – 50, 2014.

[6] C. Amato, G. Konidaris, G. Cruz, C. A. Maynor, J. P. How, and L. P. Kaelbling, "Planning for decentralized control of multiple robots under uncertainty," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1241–1248.

[7] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[8] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Wiley, 2020.

[9] R. Zou and S. Bhattacharya, "On optimal pursuit trajectories for visibility-based target-tracking game," *IEEE Transactions on Robotics*, vol. 35, no. 2, pp. 449–465, 2019.

[10] R. Murrieta-Cid, T. Muppirala, A. Sarmiento, S. Bhattacharya, and S. Hutchinson, "Surveillance strategies for a pursuer with finite sensor range," *The International Journal of Robotics Research*, vol. 26, no. 3, pp. 233–253, March 2007.

[11] S. Bhattacharya and S. Hutchinson, "A cell decomposition approach to visibility-based pursuit evasion among obstacles," *The International Journal of Robotics Research*, vol. 30, no. 14, pp. 1709–1727, Sep. 2011.

[12] J. O'Rourke, "Galleries need fewer mobile guards: A variation on Chvátal's theorem," *Geometriae Dedicata*, vol. 14, no. 3, pp. 273–283, 1983.

[13] J. Urrutia, "Art gallery and illumination problems," in *Handbook of Computational Geometry*. North-Holland, 2000, pp. 973–1027.

[14] W. pang Chin and S. Ntafos, "Optimum watchman routes," *Information Processing Letters*, vol. 28, no. 1, pp. 39 – 44, 1988.

[15] W.-P. Chin and S. Ntafos, "Shortest watchman routes in simple polygons," *Discrete & Computational Geometry*, vol. 6, no. 1, pp. 9–31, Mar 1991.

[16] G. J. Laguna and S. Bhattacharya, "Path planning with incremental roadmap update for visibility-based target tracking," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019, pp. 1159–1164.

[17] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 159–185, 2018.

[18] A. akbar Agha-mohammadi, S. Chakravorty, and N. M. Amato, "Firm: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements," *The International Journal of Robotics Research*, vol. 33, no. 2, pp. 268–304, 2014.

[19] S. M. LaValle and J. James J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[20] J. O'Rourke, *Art gallery theorems and algorithms*. New York, NY: Oxford University Press, 1987.

[21] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Dept. Comput. Sci., Iowa State Univ., Ames, IA, USA, Tech. Rep., 1998.