Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

An Information-Driven Framework for Motion Planning in Robotic Sensor Networks: Complexity and Experiments

Rafael Fierro, Silvia Ferrari, and Chenghui Cai

Abstract— **A geometric optimization based approach to deploy a mobile sensor network for the purpose of detecting and capturing mobile targets in the plane is presented in [1]. The sensing-pursuit problem is motivated by the *Marco Polo* game, in which the pursuer *Marco* must capture multiple mobile targets that are sensed intermittently, and with very limited information. In this paper we extend the results in [1] by providing (i) a complexity analysis of the proposed cell decomposition planning algorithm, and (ii) a testbed that allows to experimentally verify the applicability of the proposed pursuit methodology.**

I. INTRODUCTION

Coordination of robotic networks and sensor planning approaches have received considerable attention in recent years. Examples include landmine detection by multiple and heterogenous sensors installed on ground vehicles; sensor networks for monitoring endangered species in their natural environment; robot-installed sensors to monitor urban environments, production in manufacturing plants, and civil infrastructure; and intruder and target detection systems. These networks are expected to operate reliably in dynamic environments with little human intervention.

Several motion planning approaches have been proposed in the last few years, see for instance [2]. Most of the research relating sensor measurements to robot motion planning has focused on the effects that the uncertainty in the geometric models of the environment has on the motion strategies of the robot [3]. Hence, considerable progress has been made toward integrating sensor measurements in topological maps [4], and on planning strategies based only on partial or nondeterministic knowledge of the workspace [5]. Another line of research has investigated the extension of motion planning techniques to the problem of sensor placement for achieving coverage of unstructured environments [6]. Area coverage aims at ensuring that every point in a two-dimensional space is within the range of at least one sensor in the sensor network (e.g., [7]). Coverage control for mobile sensors has been treated in [8] using Voronoi diagrams to achieve uniform sensing performance over an area. Another well-known formulation of coverage is the art-gallery problem or line-of-sight visibility, where multiple sensors are placed such that the targets in the workspace are in the line-of-sight of at least one sensor [9].

In our previous work [1], we developed a methodology that employs cell decomposition algorithms to obtain a graph representation of the robot configuration space that is void of obstacle and enables target detections by obtaining a decomposition in which observation cells are used to represent configurations intersecting the targets. The simple philosophy behind this approach is that while the geometry of the robot must not intersect that of an obstacle to avoid collision, the geometry of the sensor field-of-view must intersect that of a target to enable a detection. At any given time, the pursuers must also detect new targets, for which there is no available information. More specifically, a set of policies that optimize a tradeoff between these multiple objectives is obtained by searching the robot configuration graph, and by performing inner-loop trajectory generation and tracking. The path obtained from the configuration graph is one that maximizes the overall probability of detection, and minimizes the distance traveled by the pursuer to detect or capture the mobile targets.

In this paper we present a complexity analysis of the cell decomposition algorithms used in the sensor path planning strategy outlined above. The complexity analysis is based on the approach described in [10], [11]. Moreover, a heterogenous multivehicle experimental testbed is briefly described and preliminary experimental results are given.

The remainder of the paper is organized as follows. We formulate the sensing-pursuit problem in Section II. Section III describes the information-driven framework that allows a group of multiple sensor agents to detect and intercept dynamic targets. Section IV contains a complexity analysis of the cell decomposition algorithm. An experimental testbed for pursuit-evasion games is outlined in Section V. Finally, we draw conclusions in Section VI.

II. PROBLEM STATEMENT AND ASSUMPTIONS

We consider a pursuit-evasion game in which N pursuers comprised of robotic sensors attempt to detect and pursue M active targets. The game takes place in a square area-of-interest $\mathcal{S} \subset \mathbb{R}^2$, with boundary $\partial\mathcal{S}$ and dimensions $L \times L$. $\mathcal{S}$ is populated by n fixed and convex obstacles $\{\mathcal{O}_1, \ldots, \mathcal{O}_n\} \subset \mathcal{S}$. The geometry of the i^{th} pursuer is assumed to be a convex polygon denoted by $\mathcal{A}_i$, with a configuration q_i that specifies its position and orientation with respect to a fixed Cartesian frame $\mathcal{F}_\mathcal{S}$. The input to pursuer i is $u_p^i = [v_p^i \quad \omega_p^i]^T$, and $u_p \in \mathcal{U} \subset \mathbb{R}^2$. The set of all pursuers in $\mathcal{S}$ is denoted by $\mathcal{P}$, and $I_\mathcal{P}$ is the index

R. Fierro is with the MARHES Lab, Electrical & Computer Engineering Department, University of New Mexico, Albuquerque, NM 87131-0001, USA rfierro@ece.unm.edu

S. Ferrari is with the Laboratory for Intelligent Systems and Controls, Department of Mechanical Engineering & Materials Science, Duke University, Durham, NC 27708-0300, USA sferrari@duke.edu

C. Cai is a Postdoctoral Research Associate of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA cc88@duke.edu

set of $\mathcal{P}$. Targets τ_j are assumed to move along straight lines. Exceptions to this rule are maneuvers used to avoid an obstacle or another target. The heading angle of a target j is denoted by θ^j_τ. The set of all targets is denoted by $\mathcal{T}$, where $I_\mathcal{T}$ is the index set of $\mathcal{T}$, and targets are assumed to enter $\mathcal{S}$ according to a random Poisson process with density λ [12], which is possibly unknown. The maximum translational speed $V_{p,\tau_{\max}}$ of all sensors and targets is known, and $V_{p_{\max}} > V_{\tau_{\max}}$ [13]. While sensors can move with any speed in $[0, V_{p_{\max}}]$, it is assumed that the speed of every target is uniformly distributed in $[V_{\tau_{\min}}, V_{\tau_{\max}}]$, with $V_{\tau_{\min}} > 0$.

In the sensing problem, the paths of the targets are represented by rays or half-lines, denoted by $\mathcal{R}^j_\theta$, that are unknown *a priori*. The sensors installed on the robotic platforms are assumed to be isotropic or omnidirectional and, therefore, their field of view is represented by a disk $\mathcal{D}_i = \mathcal{D}(p_i, r_i) \in \mathcal{S}$ with radius r_i, and centered at p_i. The sensor i installed on the robot $\mathcal{A}_i$ has the ability to *detect* the j target when $\mathcal{D}_i \cap \mathcal{R}^j_\theta \neq \varnothing$. The measurements obtained from each detection can be associated with a particular target using a data-association algorithm (such as [14]), although they may still be subject to errors and false alarms. Then, at any time t, the set of detections associated with the a target j is denoted by Z^t_j, and symbolizes all measurements of the target positions τ_j obtained since the time of the first detection t_1, e.g., $Z^t_j = \{z_j(t_1), z_j(t_2), \ldots, z_j(t_l)\}$. Since the sensors produce few individual observations for each moving target (e.g., due to their limited range) and are subject to frequent false alarms, the approach known as *track-before-detect* [15] is used, in which a set of k spatially distributed sensor detections are used to estimate the *target track*, $\mathcal{R}^j_\theta$, from Z^t_j, before declaring a positive detection. Every track may be updated every time a new measurement becomes available from its target. Once a target track has been formed from at least k sensor detections that are obtained at different moments in time, an upper-level controller declares that the target has been positively detected, and deploys a pursuer to capture it. The inputs to the pursuers take into account the information available from all targets, $Z^t = \{Z^t_j \mid j \in I_\mathcal{T}\}$, in order to optimize their sensing and pursuit performance.

In the problem considered here no communication between targets and pursuers takes place, but the pursuers may obtain only position information about the targets when they enter their field of view. Based on the previous discussion, the sensing-pursuit problem can be stated as follows:

Problem 2.1: Given a set $\mathcal{P}$ of N pursuers and a set $\mathcal{T}$ of M targets moving within a specified game area $\mathcal{S}$, find a set of policies $u^i_p = c^i(q_i, Z^t) \in \mathcal{U}\ \forall i \in I_\mathcal{P}$ which maximizes the total sensing reward, and minimizes the total time required to capture targets in $\mathcal{T}$ that have been positively detected.

To complete the formulation of above problem, we define the sensing reward in terms of the probability of detection, as explained in Section III. Also, sensors operate in one of two modes, *detection* or *pursuit*, depending on whether their primary objective is to detect targets or to capture them. Also, we assume that sensors have sufficient processing capabilities to determine the time and position of a detection event from their raw measurements. In this problem, target tracks are classified based on the following definitions:

Definition 2.2: An unobserved track is the path of a target j for which there are no detections at the present time, t, thus $Z^t_j = \emptyset$.

Definition 2.3: A partially-observed track is the path of a target that is estimated from $1 < l < k$ individual sensor detections obtained up to the present time, t, i.e., $Z^t_j = \{z_j(t_1), \ldots, z_j(t_l)\}$.

Definition 2.4: A fully-observed track is the path of a target that is estimated from at least $k > 2$ individual sensor detections obtained up to the present time, t, i.e., $Z^t_j = \{z_j(t_1), \ldots, z_j(t_m)\}$, where $t_m \geq t_k$.

The parameter k is chosen by the user based on the reliability of the sensors detections and on the cost associated with deploying a pursuer to capture the target. For instance, in [15] it was found that from a geometric point of view $k = 3$ is a convenient number of detections for estimating a track in the absence of false alarms. Only after a track is fully-observed, the target is considered to be *positively detected*. Then, the estimated track is used by an upper-level controller to decide which pursuer to deploy and switch to pursuit mode, and to compute a pursuit strategy online.

The objectives of the sensors in detection mode are to (i) avoid obstacles; (ii) maximize the probability of cooperatively detecting unobserved tracks; and (iii) maximize the probability of detecting p partially-observed tracks $\{\mathcal{R}^1_\theta, \ldots, \mathcal{R}^p_\theta\}$. On the other hand, the objectives of a sensor in pursuit mode are to (i) avoid obstacles; and (ii) minimize the time required to capture a positively detected target j.

The following section describes a methodology for planning the motions of the pursuers, in order to meet all of the above objectives.

III. INFORMATION-DRIVEN SENSOR PLANNING

The methodology described in this section aims at developing policies that meet multiple sensing and motion objectives for sensors in detection or pursuit mode. The primary purpose for planning the motion of the pursuers in detection mode is to obtain measurements from the targets. But since the target tracks may be unknown (unobserved) or uncertain (partially-observed), the sensors motion cannot be planned using classical motion planning objectives, such as, minimizing distance and reaching a final configuration [16], [17]. In fact, the positions and field-of-views of all sensors must be taken into account to plan the motion of a sensor that performs coordinated detections. Let $\mathcal{C}_{free}$ denote the robotic sensors' configuration space that is free of obstacles, thus:

Definition 3.1: A void cell is a convex polygon $\kappa \subset \mathcal{C}_{free}$ with the property that for every configuration $q_i \in \kappa$ the sensor i has zero probability of detecting a partially-observed target.

In order to account for the geometries and dynamics of the sensors's field of view and of those of the targets, we also introduce the following definition:

Definition 3.2: An observation cell is a convex polygon $\underline{\kappa} \subset \mathcal{C}_{free}$ with the property that for every configuration $q_i \in \kappa$ the sensor i has a non-zero probability of detecting a partially-observed target.

Void and observation cells are determined such that an obstacle-free sensor path can be easily computed between any two configurations inside each cell. Furthermore, two cells are said to be *adjacent* if they share a common boundary and, therefore, the sensor can move between them without colliding with the obstacles. Typically, all cells are computed such that they do not overlap. In Section III-A, a method for obtaining these cells for the system in Problem 2.1 is presented. Subsequently, they can be used to obtain the following graph:

Definition 3.3: A connectivity graph, $\mathcal{G}$, is an undirected graph where the nodes represent either an observation cell or a void cell, and two nodes in $\mathcal{G}$ are connected by an arc if and only if the corresponding cells are adjacent.

The purpose for deploying sensors in detection mode is to detect unobserved and partially-observed target tracks. Thus, the sensing objectives are expressed in terms of a reward function that represents the improvement in the overall probability of detection that would be obtained by moving from a configuration in one cell, $q_i \in \kappa_l$, to a configuration in an adjacent cell, $q_i \in \kappa_\iota$,

$$R(\kappa_l, \kappa_\iota) = P_{\mathcal{R}}(\kappa_\iota) + \Delta P^k_{\mathcal{S}}(\kappa_l, \kappa_\iota), \quad (1)$$

where $P_{\mathcal{R}}$ is the probability of detecting a target with a partially-observed track, and $\Delta P^k_{\mathcal{S}}$ is the gain in the probability of cooperatively detecting unobserved tracks. Since these quantities may vary slightly within each cell, they are computed in reference to the geometric centroid, where $\bar{q}_i$ denotes the centroid of cell κ_i. As illustrated in the following section, the value of the reward function is attached to every arc in $\mathcal{G}$. Then, the optimal sequence of cells or *channel* that maximizes the total reward of a sensor in detection mode is,

$$\mu^* \equiv \{\kappa_0, \ldots, \kappa_f\}^* = \arg\max_{\mu} \sum_{(\kappa_l, \kappa_\iota) \in \mu} R(\kappa_l, \kappa_\iota), \quad (2)$$

where, κ_f is chosen as the observation cell with the highest cumulative probability in $\mathcal{G}$, i.e., $\kappa_f = \arg\max_{\underline{\kappa}_i}(P_{\mathcal{R}}(\underline{\kappa}_i) + P^k_{\mathcal{S}}(\underline{\kappa}_i))$. Once the connectivity graph $\mathcal{G}$ has been obtained, the optimal channel μ^* is computed using the A* graph searching algorithm [16]. Once μ^*_i is determined for sensor i, it is mapped into a set of waypoints, which in turn are used by a trajectory generator and trajectory tracking controller to determine its policy $u^i_p = c^i(q_i, Z^t)$. The values of the reward function (1) attached to the arcs, κ_0, and κ_f are different for each sensor. Therefore, the A* algorithm must be run during every round of the game which involves moving a sensor in detection mode.

A. Probability of detection for partially-observed tracks

The partially-observed tracks are viewed as an opportunity for obtaining additional measurements before investing in the costly resources needed to capture a target. In order to account for the geometry of the sensor field-of-view $\mathcal{D}_i$, the platform $\mathcal{A}_i$, and the target track $\mathcal{R}^j_\theta$, we present an approach motivated by cell decomposition algorithms [16]. The simple philosophy behind this approach is that, in sensor planning problems, targets play a role opposite to that of obstacles in robot motion planning. While in robot motion planning the geometry of the robot must avoid intersecting that of any obstacle, in sensor planning the geometry of the sensor field-of-view must attempt to intersect that of the target.

Let $\mathcal{F}_{\mathcal{A}_i}$ denote a moving Cartesian frame embedded in $\mathcal{A}_i$. The configuration q_i specifies the position and orientation of $\mathcal{F}_{\mathcal{A}_i}$ with respect to the inertial frame $\mathcal{F}_{\mathcal{S}}$. If we assume that $\mathcal{D}_i$ and $\mathcal{A}_i$ are both rigid, then q_i also specifies the position of every point in $\mathcal{D}_i$ (or $\mathcal{A}_i$) relative to $\mathcal{F}_{\mathcal{S}}$. Using the latest estimate of a partially-observed track, it is possible to identify the subset of $\mathcal{S}$ where the sensors may obtain target measurements:

Definition 3.4 (C-target): The target track $\mathcal{R}^j_\theta$ in $\mathcal{S}$ maps in the i^{th} sensor configuration space $\mathcal{C}$ to the C-target region $\mathcal{CR}_j = \{q_i \in \mathcal{C} \mid \mathcal{D}_i \cap \mathcal{R}_j \neq \varnothing,\ i \in I_{\mathcal{P}},\ j \in I_{\mathcal{T}}\}$.

The boundary of a C-target is the curve followed by the origin of $\mathcal{F}_{\mathcal{A}_i}$ when $\mathcal{D}_i$ slides in contact with the boundary of $\mathcal{R}^j_\theta$. With the assumed robot and sensor geometries, the C-target boundaries are obtained by growing $\mathcal{R}^j_\theta$ isotropically by the radius r_i within $\mathcal{S}$. C-obstacles are similarly defined [16] and are used together with the C-targets introduced above to obtain the connectivity graph of each sensor.

Let $\mathcal{CO}_k$ denote the C-obstacle obtained from the k^{th} obstacle in the game area, $\mathcal{O}_k \subset \mathcal{S}$. In obstacle avoidance algorithms, the obstacle-free configuration space $\mathcal{C}^i_{free} = \mathcal{C} \setminus \bigcup_{k=1}^{n} \mathcal{CO}_k$ is decomposed into a finite set of cells, $\{\kappa_1, \ldots, \kappa_f\}$, within which a path free of obstacles can be easily generated. In order to obtain a decomposition that includes observation cells (Definition 3.2), we present the following method:

(I) Decompose the configuration space that is void of any C-obstacles or C-targets and is defined as $\mathcal{C}^i_{void} = \mathcal{C} \setminus \{\bigcup_{j=1}^{n} \mathcal{CO}_k \ \cap\ \bigcup_{i=1}^{p} \mathcal{CR}_j\}$,

(II) Decompose each obstacle-free C-target,

$$\mathcal{CR}_j \ \setminus \bigcup_{j=1}^{n} \mathcal{CO}_k, \ \ j = 1, \ldots, p$$

thereby obtaining the set of observation cells.

(III) Construct a connectivity graph $\mathcal{G}$ using both void (I) and observation cells (II).

When the C-targets are grown isotropically by a disk (Fig. 1), the decomposition may involve generalized polygons. A sweeping-line algorithm can be used to decompose a non-convex generalized polygon with ν vertices into $O(\nu)$ convex generalized polygons in $O(\nu \log \nu)$ time (see Section 5.1 in [16]). Alternatively, the pill-shape C-targets can be approximated by a convex polygon. An illustrative example of workspace and corresponding cell decomposition is shown in Fig. 1. The connectivity graph constructed using this cell decomposition is illustrated in Fig. 2, where the observation cells are shown in grey and the void cells are white. Each node in the connectivity graph corresponds to one polygonal cell in Fig. 1, where the cells are numbered from left to right and from top to bottom.

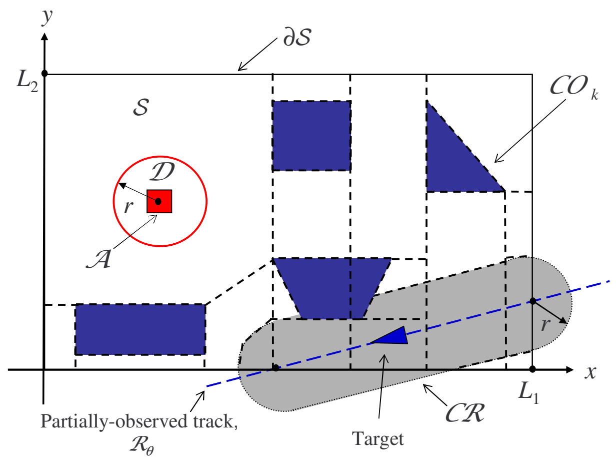
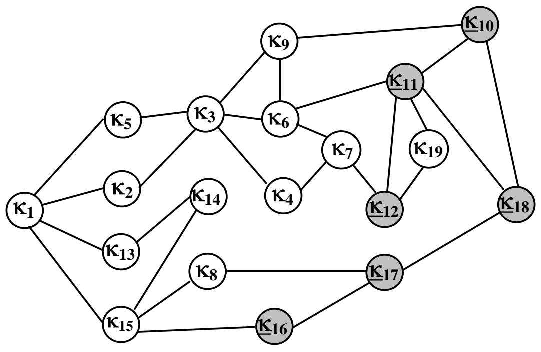


Fig. 1: Example of cell decomposition (dashed lines) for a workspace with four C-obstacles.



κ_1
κ_2
κ_3
κ_4
κ_5
κ_6
κ_7
κ_8
κ_9
$\underline{\kappa}_{10}$
$\underline{\kappa}_{11}$
$\underline{\kappa}_{12}$
κ_{13}
κ_{14}
κ_{15}
$\underline{\kappa}_{16}$
$\underline{\kappa}_{17}$
$\underline{\kappa}_{18}$
κ_{19}

Fig. 2: Connectivity graph obtained from the cell decomposition in Fig. 1.

Now, we are ready to show one information benefit function, $P_{\mathcal{R}}$, used to compute the reward (1). Suppose $\underline{\kappa}_l$ is one of the observation cells that are obtained from the decomposition of the j^{th} C-target: $\underline{\kappa}_l \subset \mathcal{CR}_j$. Then, one benefit of visiting the l^{th} cell in $\mathcal{G}$ is the probability of detecting the target j

$$P_{\mathcal{R}}(\underline{\kappa}_l \subset \mathcal{CR}_j) = \Pr\{D_{ji} \mid e_{ji} \leq r_i\}, \tag{3}$$

where D_{ji} represents the event that the i^{th} sensor reports a detection when the j^{th} target comes within its detection range, and e_{ji} denotes the Euclidean distance from the j^{th} target position to the sensor i^{th}. In this paper, $P_{\mathcal{R}}$ is assumed to be uniform over $\mathcal{CR}_j$ for simplicity, and when a cell is void $P_{\mathcal{R}}(\kappa_l) = 0$ since the sensor field-of-view will not intersect any of the p partially-observed tracks. In general, it can be estimated from knowledge of the measurement process and can be made dependent on time and on the distance from the target [18]. The probability of cooperatively detecting unobserved tracks is discussed in the following section.

B. Search area coverage

At any given time, the team of sensors must also detect unobserved tracks. These tracks may belong to targets that only recently entered the search area $\mathcal{S}$ or that have been previously missed. Since the targets are always in motion, maximizing area coverage or other coverage formulations may not lead to effective cooperative detections. In proximity sensor systems, for example, detections take place at different moments in time anywhere in $\mathcal{S}$. Once a sufficient number of detections k has been obtained from the same target and can be used to form a feasible track a positive detection can be declared with confidence. The quality of service of an omnidirectional sensor network performing cooperative detections of moving targets is referred to as track coverage [19]. Since track coverage of a sensor network can be assessed without any prior knowledge of the target tracks, it is used to determine the sensors' ability to detect unobserved tracks. The details of computing the gain in the probability of detecting unobserved tracks $\Delta P_{\mathcal{S}}^k$ are presented in [1], [19].

One goal of the sensors in detection mode is to detect unobserved tracks traversing $\mathcal{S}$. When a sensor may move to a cell κ_l, the network configuration can be approximated by $\mathcal{X}_l = \{p_1, \ldots, p_i \subset \bar{q}_l, \ldots, p_N\}$, letting the center of the sensors' field-of-view, $\mathcal{D}_i$, coincide with the centroid $\bar{q}_l$ of κ_l. Thus, the gain in probability of detection for unobserved tracks that is associated with moving between two nodes $\kappa_l \to \kappa_\iota$ in $\mathcal{G}$ is

$$\Delta P_{\mathcal{S}}^k(\kappa_l, \kappa_\iota) \equiv P_{\mathcal{S}}^k(\mathcal{X}_\iota) - P_{\mathcal{S}}^k(\mathcal{X}_l) \tag{4}$$

An effective control approach by which sensors in pursuit mode capture and intercept targets whose tracks have been fully observed and, thus, have been declared positively detected is presented in [20].

IV. COMPLEXITY ANALYSIS

Previous work on the correctness and complexity of pursuit-evasion games has focused on graphs, in which one or more pursuers attempt to capture one target by moving between adjacent nodes in a graph (see [13], [21], [22] for a comprehensive review). In these problem formulations, the sensing ability and field-of-views of the pursuers are not taken into account, and the pursuit strategies consist of randomized searches on the graph, because the pursuer cannot see the evader until the former is caught. Also, only one evader who may be restricted or unrestricted to the graph is considered during each game. By computing the connectivity graph by the methodology in Section III-A, these results could potentially be extended to the pursuit-evasion game in Problem 2.1. For example, if the strategy in [21] is implemented for one pursuer and one evader ($N = M = 1$), then pursuer captures the evader on an n-node cycle with probability at least $\Omega(1/\log(n_{\mathcal{G}}))$, and the game ends in $O(n_{\mathcal{G}} \cdot \log(\text{diam}(\mathcal{G})))$ time, where $n_{\mathcal{G}}$ is the number of nodes in $\mathcal{G}$ and $\text{diam}(\mathcal{G})$ is the diameter of the graph. However, by not taking into account the sensing ability of the pursuers and the consequent knowledge of fully-observed tracks (i.e., the presence of observation cells), these strategies are not very effective at capturing multiple evaders in large game areas. In these applications, $n_{\mathcal{G}}$ and $\text{diam}(\mathcal{G})$ are very large, therefore the probability of capturing the evaders can become very small and, at the same time, the game end time $O(M \cdot n_{\mathcal{G}} \cdot \log(\text{diam}(\mathcal{G})))$ can become very large.

The correctness and game-end time for the strategy presented in Section III are analyzed by assuming that the time required to maneuver around obstacles or to turn are negligibly small compared to the duration of the game. Let $\bar{(\cdot)}$ denote the expected value (or mean), and $\lfloor \cdot \rfloor$ denote the floor function. Then, the performance of the sensor network depends on the dimension of the game area L, the number of sensors N, the number of required detections $k > 2$, and the field-of-view radius r_i, which here is assumed constant $r_i = r$, $\forall i$ for simplicity, as summarized by the following result.

Theorem 4.1: The pursuit-evasion game in Problem 2.1 is guaranteed to terminate provided,

$$N \geq N_{\min} = \frac{1}{2}\left[\left\lfloor \frac{2L}{r} \right\rfloor + k \left| \left\lfloor \frac{2L}{r} \right\rfloor - k + 2 \right| \right], \quad (5)$$

and requires a time $t_f \leq T_u$ where

$$\begin{aligned} T_u &= \frac{e(M!)^{1/M}}{\lambda} + \frac{(\sqrt{2}L - 2r)}{V_{\tau_{\min}}} \quad (6)\\ &+ \left[\left\lfloor \frac{(k-2)M}{N} \right\rfloor + 1\right] \frac{(\sqrt{2}L - r)}{\bar{V}_p} + \frac{r}{(V_{p_{\max}}^2 - \bar{V}_\tau^2)} \\ &+ \left\lfloor \frac{M}{N} \right\rfloor \frac{(\bar{V}_\tau + \sqrt{2V_{p_{\max}}^2 - \bar{V}_\tau^2})}{(V_{p_{\max}}^2 - \bar{V}_\tau^2)^2} L, \end{aligned}$$

to capture all M targets in $\mathcal{T}$. If the network contains at least

$$\begin{aligned} N_\ell &= \frac{1}{2}\left[\ell \left\lfloor \frac{2L}{r} \right\rfloor\right] - 4\ell(\ell-1) + (k-2)M \quad (7)\\ &+ \left| \ell \left\lfloor \frac{2L}{r} \right\rfloor - 4\ell(\ell-1) - (k-2)M \right|, \end{aligned}$$

sensors, with $\ell = 1, \ldots, \lfloor L/4r \rfloor$, then all targets in $\mathcal{T}$ can be captured in a time $t_f \leq T_\ell$, where

$$\begin{aligned} T_\ell &= \frac{e(M!)^{1/M}}{\lambda} + \frac{1}{V_{\tau_{\min}}}\left\{\frac{\sqrt{2}}{2}L - 2\sqrt{2}r(\ell-1)\right\} \\ &+ \frac{1}{V_{\tau_{\min}}}\left| 2r[1 + \sqrt{2}(\ell-1)] - \frac{\sqrt{2}}{2}L \right| \\ &+ \frac{(\sqrt{2}L - r)}{\bar{V}_p} + \frac{r}{(V_{p_{\max}} - \bar{V}_\tau)}, \quad (8) \end{aligned}$$

and the game terminates in $t_f \leq T_\ell \leq T_u$, where $T_\ell = T_u$ when $\ell = 1$ and $k = 3$.

Based on the above result, $N_{\min}$ is the minimum number of sensors needed to guarantee that the game will end in less than T_u time. But, if more sensors can be utilized, then N can be increased according to (7) to decrease the maximum time required to end the game, as shown by (8).

In Problem 2.1, a round is defined as the deployment of one sensor in either detection or pursuit mode, and it is initiated based on the measurement set Z^t when a new target track becomes either partially observed or fully observed. Thus, the computational complexity of the methodology in Section III is assessed based on the calculations required by each round. Let $n_{e_{\mathcal{O}}}$ denote the number of edges required to describe all n obstacles in $\mathcal{S}$, $n_{e_{\mathcal{R}}} = 2p$ denote the number of edges required to describe all p tracks that have been partially observed up to the present time. Then, if $n_{\underline{\mathcal{K}}}$ and $n_{\mathcal{G}_a}$ are the number of observation cells and the number of arcs in $\mathcal{G}$, respectively, δb is a constant interval to discretize $\partial\mathcal{S}$, and $n_\delta \equiv L/\delta b$, the following result can be obtained for the method in Section III.

Theorem 4.2: In every round of the pursuit-evasion game in Problem 2.1, the running time required to deploy a sensor in detection mode is,

$$\begin{aligned} \Gamma_d &= O(n_{e_{(\mathcal{O}+\mathcal{R})}} \log n_{e_{(\mathcal{O}+\mathcal{R})}} + n_{e_{\mathcal{R}}} n_{e_{\mathcal{O}}} \log n_{e_{\mathcal{O}}}) \\ &+ O(n_{\underline{\mathcal{K}}} n_\delta m(k + \log m)) + O(n_{\mathcal{G}}^2 + n_{\mathcal{G}_a}), \quad (9) \end{aligned}$$

where $n_{e_{(\mathcal{O}+\mathcal{R})}} := n_{e_{\mathcal{O}}} + n_{e_{\mathcal{R}}}$ and $m = \begin{pmatrix} N \\ k \end{pmatrix}$ is given by the binomial coefficient, and the running time required to deploy a sensor in pursuit mode is,

$$\Gamma_p = O((n_{\mathcal{G}} + n_{\mathcal{G}_a}) \log_2 n_{\mathcal{G}}). \quad (10)$$

Clearly, depending on the characteristics of the robotic sensors and of the workspace $\mathcal{S}$ only one of the three terms in (9) will dominate over the others, providing the overall running time complexity of the detection round. Due to space limitations proofs of the theorems are omitted here.

V. MULTIVEHICLE PURSUIT-EVASION EXPERIMENTS

This section describes a pursuit-evasion experiment using several robots from the heterogeneous MARHES multivehicle platform [23]. Essentially, the experiment uses cameras which are placed at known locations within a secure area to monitor for intruding robots. When an intruder robot is detected within the secure area, a pursuer robot is dispatched to intercept the intruder.

1) Experimental Objectives: The goal of this experiment is to use pursuer robots to intercept intruder that are being tracked with a set of cameras. In the remainder of this subsubsection we formally state our experimental objectives and assumptions.

At the beginning of each experiment, N pursuer robots are placed with known location and orientation $q_i(0) = [x_i(0), y_i(0), \theta_i(0)]$ along the boundary $\partial\mathcal{S}$. These robots are nonholonomic vehicles with maximum linear velocity $V_{p\max}$ and angular velocity $\Omega_{p\max}$, respectively. At any given time during the experiment, a pursuer robot will be in one of two modes. Pursuer robots in *pursuit* mode have been assigned a target location but have not yet reached that location. When a pursuer robot has not been assigned a target or has reached its last target, it enters *standby* mode. For this experiment, invading or target robots are assumed to travel in straight lines with fixed velocities. Thus, these robots can be completely specified using four parameters including $p_{\mathcal{T}}(0) = [x_{\mathcal{T}}(0), y_{\mathcal{T}}(0), \theta_{\mathcal{T}}]$ and $v_{\mathcal{T}}$, where $p_{\mathcal{T}}(0)$ represents the initial position and orientation of the target, and $v_{\mathcal{T}}$ is the target's speed. In order to ensure that the pursuer robots can capture the targets, we assume $V_{p\max} > v_e$ for all pursuers $\mathcal{P}$ and intruders $\mathcal{T}$.

In order to monitor for targets, a set $\mathcal{Q}$ of cameras are placed along the boundary $\partial\mathcal{S}$ at known locations $p_c(0) = [x_c(0), y_c(0), \theta_c(0)]$. In addition, each camera's depth of view and field of view are assumed to be known. When combined with the camera's orientation, these two values can be used to determine a cone of visibility $\mathcal{V} \in \mathbb{R}^2$. Each camera's cone of visibility should lie at least partially within $\mathcal{S}$. We assume that a method exits to uniquely identify target robots that are within the cone of visibility and to localize them with respect to a fixed frame of reference which is attached to the camera. Based on knowledge of p_c, a coordinate transformation can then be applied to convert these measurements to a global frame $\mathcal{F}_{\mathcal{S}}$. Each of the cameras can be configured in two modes including *static sensor mode* and *pursuer camera mode*. A pursuer camera is attached to one of the pursuer robots and thus moves with the robot. Static cameras are fixed at their initial positions during the entire experiment. We can now state the experimental objective formally as follows.

Objective 1: Given a set of cameras $\mathcal{Q}$ and pursuer robots $\mathcal{P}$, monitor the region $\mathcal{S} \bigcap (\bigcup_i \mathcal{V}_i)$ for target robots. Observe each target $\tau_i \in \mathcal{T}$ and continue taking measurements until its path and velocity can be estimated. Once target τ_i's path has been identified (that is, $p^i_{\mathcal{T}}(0)$ and $v^i_{\mathcal{T}}$ are known), send the a pursuer robot from the set of pursuers currently in standby mode to intercept. When multiple possible pursuers could be chosen, chose the one which can reach the intruder fastest.

To facilitate implementation of the system's monitoring and pursuit abilities, the system architecture is organized into three key components including camera modules, robot modules, and a centralized coordinator. Since the target robots are traveling in straight lines, their implementation is straightforward and will not be discussed further here. A diagram of the architecture for monitoring and pursuit is shown in Figure 3.

The central component of the system architecture is a coordinator application which consists of three main components. First, the camera control client is responsible for connecting to each of the camera servers and downloading data about detected targets. Second, the robot control client submodule, is responsible for converting the detection database into estimates of each target's path and velocity. Once these calculations are complete, the robot control client submodule assigns a pursuer robot to intercept the target. The final submodule is the heads up display (HUD). This submodule provides a graphical display of the experiment's progress.

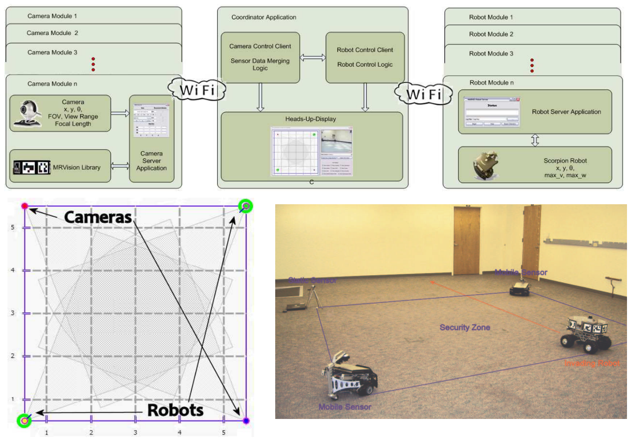


Fig. 3: The monitoring and pursuit architecture shown in this figure consists of three key components including camera modules, robot modules, and a centralized coordinator.

Feedback is provided in two forms. First, the user can select from any one of the camera modules and view that camera's video over the wireless network. The second component is a two dimensional drawing of the experiment's state. The secure area is drawn using dark blue lines. Additionally the positions and orientations of all cameras and all pursuers are indicated on the display. Each camera's cone of visibility $\mathcal{V}$ is displayed as well so that the coverage of the security area can be visualized. As detections are made, they are shown using color coded dots to indicate which camera they are from. Once an target's path is identified, the track is drawn on the display as well. At this point a pursuer will be assigned for interception and the HUD shows its anticipated trajectory. Finally, as the robot moves its odometer is recorded and the actual trajectory is displayed as well.

2) Experimental Results: Using the experimental setup discussed above, two target robots were driven simultaneously through the security region. In this experiment, the target speeds were set to approximately one-half of the maximum linear velocity of the pursuer robots. Figure 4 shows snapshots of the HUD at several key points during the experiment. The initial setup of the experiment is shown in (a). As the two invaders begin to move through the security region, detection points start to appear as shown in (b). Since the two intruders have different NameTags, the coordinator can distinguish between them. Detections of one invader are outlined in yellow while detections of the other are in red. After a fixed number of detections occur, the coordinator estimates the intruder's path (red line) and velocity as shown in (c). Based on these values, a red circle which indicates the intruders estimated position and orientation is drawn as well. The robot in the lower left corner is assigned to intercept and the green line represents the anticipated trajectory. Soon afterward, the second invader's path is identified as well and the yellow path is drawn as shown in (d). The one remaining pursuer robot in standby mode is assigned to intercept and again the anticipated trajectory is shown in green. In (e) we see that both the robots have moved and the leftmost pursuer has captured its target (It appears to be about half a meter away due to the physical sizes of the robots). The purple lines indicate the robots' true paths according to their odometers. Finally (f) shows the system state once the experiment has completed. Overall, we see that the coordinator was able to successfully identify the invaders' tracks and assign pursuer robots to intercept them.

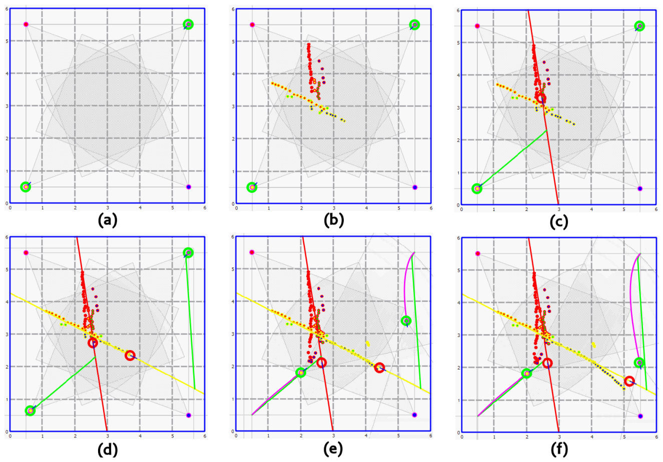


Fig. 4: This figure shows a set of snapshots from the HUD during one run of the pursuit evasion experiment developed in this section. Initially the robots and cameras are positioned as in (a). As the experiment progresses, the intruders are detected as in (b) and eventually the tracks are identified as in (c) and (d). Finally, in (e) and (f) we see the pursuer robots moving to intercept their respective targets.

We are currently working on enhancing the system's accuracy and robustness by rectifying the camera images to account for lens distortion.

VI. CONCLUSIONS

This paper presents a novel information-driven framework involving multiple robotic platforms that seek to detect and intercept mobile targets. Multiple objectives, such as the probability of detecting unobserved tracks, obstacle-avoidance, and the profit of information associated with partially-observed targets are addressed using an optimization based geometric approach. The complexity of the cell decomposition planning algorithms is formally analyzed. The future work of this approach will include fully implementing the methodology on the testbed shown in Fig. 3.

VII. ACKNOWLEDGMENTS

The work of R. Fierro is supported by NSF grants ECCS CAREER #0811347, IIS #0812338, CNS #0709329, and DOE NNSA Grant DE-FG52-04NA25590 (through the UNM Manufacturing Engineering Program). The work of S. Ferrari is supported in part by the Office of Naval Research Young Investigator Program (Code 321). We thank James McClintock for his assistance in obtaining the experimental results reported herein.

REFERENCES

[1] S. Ferrari, C. Cai, R. Fierro, and B. Perteet, "A geometric optimization approach to detecting and intercepting dynamic targets," in *Proceedings of the American Control Conference*, New York City, July 2007, pp. 5316–5321.

[2] A. Ganguli, J. Cortés, and F. Bullo, "Maximizing visibility in nonconvex polygons: Nonsmooth analysis and gradient algorithm design," *SIAM Journal on Control and Optimization*, vol. 45, no. 5, pp. 1657–1679, 2006.

[3] S. Koenig, C. Tovey, and Y. Smirnov, "Performance bounds for planning in unknown terrain," *Artificial Intelligence*, vol. 147, pp. 253–279, 2003.

[4] S. Thurn, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, pp. 21–71, 1998.

[5] A. Lazanas and J. C. Latombe, "Motion planning with uncertainty - a landmark approach," *Artificial Intelligence*, vol. 76, pp. 287–317, 1995.

[6] M. Qian and S. Ferrari, "Probabilistic deployment for multiple sensor systems," *Proc. SPIE Symposium on Smart Structures and Materials*, 2005.

[7] V. Isler, S. Khanna, and K. Daniilidis, "Sampling based sensor-network deployment," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 100, pp. 1780–1785, 2004.

[8] J. Cortés, S. Martínez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, April 2004.

[9] T. Shermer, "Recent results in art galleries," *Proc. IEEE*, vol. 80, no. 9, pp. 1384–1399, 1992.

[10] C. Cai and S. Ferrari, "Information-driven sensor path planning by approximate cell decomposition," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. in press, 2008. [Online]. Available: http://fred.mems.duke.edu/silvia.ferrari/SMCDeminingarticle.pdf

[11] S. Ferrari and C. Cai, "Information-driven search strategies in the board game of CLUE®," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. in press, 2008. [Online]. Available: http://fred.mems.duke.edu/silvia.ferrari/SMCCLUEarticle.pdf

[12] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*. New York, NY: McGraw-Hill, 2002.

[13] V. Isler, S. Kannan, and S. Khanna, "Randomized pursuit-evasion in a polygonal environment," *IEEE Trans. on Robotics*, vol. 21, no. 5, pp. 875–884, 2005.

[14] A. Gad, F. Majdi, and M. Farooq, "A comparison of data association techniques for target tracking in clutter," in *Proceedings of the Fifth International Conference on Information Fusion*, vol. 2, 2002, pp. 1126–1133.

[15] T. A. Wettergren, R. L. Streit, and J. R. Short, "Tracking with distributed sets of proximity sensors using geometric invariants." *IEEE Trans. Aerospace and Electronic Systems*, vol. 40, no. 4, pp. 1366–1374, 2004.

[16] J. C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[17] M. K. Habib and H. Asama, "Efficient method to generate collision free paths for autonomous mobile robots based on new free space structuring approach," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Osaka, Japan, 1991, pp. 563–567.

[18] S. Ferrari and A. Vaghi, "Demining sensor modeling and feature-level fusion by bayesian networks," *IEEE Sensors*, vol. 6, pp. 471–483, 2006.

[19] S. Ferrari, "Track coverage in sensor networks," in *Proceedings of the American Control Conference*, Minneapolis, MN, June 14-16 2006, pp. 2053–2059.

[20] B. Perteet, J. McClintock, and R. Fierro, "A multi-vehicle framework for the development of robotic games: The Marco Polo case," in *IEEE Int. Conf. on Robotics and Automation*, Rome, Italy, April 10-14 2007, pp. 3717–3722.

[21] M. Adler, H. Räcke, N. Sivadasan, C. Sohler, and B. Vöcking, "Randomized pursuit-evasion in graphs," *Combinatorics, Probability and Computing*, vol. 12, pp. 225–244, 2003.

[22] T. D. Parson, "Pursuit-evasion in a graph," in *Lecture Notes on Mathematics*, Y. Alavi and D. Lick, Eds., 1976, pp. 426–441.

[23] D. Cruz, J. McClintock, B. Perteet, O. Orqueda, Y. Cao, and R. Fierro, "Decentralized cooperative control: A multivehicle platform for research in networked embedded systems," *IEEE Control Systems Magazine*, vol. 27, no. 3, pp. 58–78, June 2007.