17th Mediterranean Conference on Control & Automation
Makedonia Palace, Thessaloniki, Greece
June 24 - 26, 2009

Cooperative Multi-Target Tracking via Hybrid Modeling and Geometric Optimization

Domagoj Tolic*, Rafael Fierro* and Silvia Ferrari†
* MARHES Lab, Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, New Mexico 87131–0001, {dtolic, rfierro}@ece.unm.edu
† Department of Mechanical Engineering and Materials Science, Duke University, Durham, North Carolina 27708-0005, sferrari@duke.edu

***Abstract*—In this paper, we present a stochastic hybrid model of mobile networks able to encompass a large variety of multi-agent problems and phenomena. The model is applied to a case study where a heterogeneous mobile sensor network cooperatively detects and tracks mobile targets in the plane based on intermittent observations. When these observations form a satisfactory target trajectory, a mobile sensor is switched to pursuit mode and deployed to capture the target in minimum time. The mobile sensor network consists of a set of robotic sensors modeled as hybrid systems with processing capabilities. Since the sensors are installed on robotic platforms and have limited range, the geometry of the mobile sensors' field-of-view plays a critical role in motion planning and obstacle avoidance. The cost of operating the sensors is determined from the geometric properties of the network, its workspace and the probability of target detection. Simulation results verify the validity of the developed model and tracking methodology.**

Fig. 1. A cooperation of UAVs and UGVs.

I. INTRODUCTION

Every day, we witness numerous achievements and improvements in both technology and science. The field of robotics is not an exception. Technology pushes hardware limits further providing processors decreasing in size and increasing in performance, more advanced robots and sensors. As a consequence, networks of commercially available UAVs and UGVs[1] are capable of successfully resolving complicated problems such as search and rescue missions, monitoring urban environments or endangered species, landmine or intruder detection, and pursuit-evasion problems. At the same time, science invents methods and techniques to solve these challenging tasks more efficiently. This means better coordination of large heterogeneous robot networks, improvements in planning, sensing and estimation requirements along with higher flexibility, robustness and fault tolerance of the networks.

Considerable number of methodologies for coordination of robotic networks and sensor planning approaches have been proposed in recent years. Distributed control of synchronous robotic networks with an emphasis on communication protocols and geometric notions relevant in motion coordination are described in [1]. An investigation of maintaining connectivity of a dynamic multi-agent network including hybrid modeling is discussed in [2]. An overview of stochastic hybrid models is given in [3]. Planning algorithms are thoroughly explained in [4], while [5] describes autonomous mobile robots from sensing, decision making and application perspectives. A hybrid modeling framework for robust maneuver-based motion planning algorithms for nonlinear systems with symmetries is proposed in [6]. Cell decomposition approaches are covered in [7], whereas a specific case of cell decomposition is implemented in [8]. All aforementioned approaches focus only on certain problems in modeling multi-agent cooperation. However, we need a broad, yet simple enough, model of mobile multi-agent networks for our work.

In this paper, we present a comprehensive hybrid network model able to capture a wide range of multi-agent problems. By applying the model to a multi-target tracking case study, we demonstrate its versatility and flexibility. The multi-target tracking case study is motivated by the Marco Polo game (first introduced in [9]) where a network of mobile robotic sensors must track and capture mobile targets based on the information obtained through cooperative detections of the sensors. This pursuit-evasion game combines cooperative multi-target tracking, distributed estimation, intermittent communication and geometric properties of the sensor network. Specifically, we extend this previous work in order to consider more realistic scenarios. An illustration of an multi-target environment is shown in Figure 1 where aerial and ground sensors are used to detect and capture mobile targets in the plane. Numerous simulations are successfully carried out in order to verify the model.

The remainder of the paper is organized as follows. In Section II we state the problem and assumptions considered

[1] *Unmanned Aerial Vehicles and Unmanned Ground Vehicles, respectively.*

herein. A detailed analysis of concepts and methods stated in Section II is given in Section III. Mathematical details of the developed hybrid model are conveyed in the first subsection of Section III. The second part of the section is reserved for concepts and definitions regarding the application of the hybrid model and methods used to solve the multi-target tracking case study. In Section IV we further investigate specifics of the applied hybrid model. Simulations and numerical results are provided in Section V. Finally, we draw conclusions in Section VI.

II. Problem Statement and Assumptions

The problem considered in this paper is stated here. *Given a heterogeneous 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 control policies of sensors which maximizes the total sensing reward, and minimizes the total time required to capture targets in $\mathcal{T}$ that have been positively detected. 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 partially-observed tracks. 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, based on its fully-observed track and the first $k-1$ detections where $k \in \mathbb{N}$.*

In this paper, targets move in piece-wise straight fashion with uniformly distributed orientations. We require that the scalar product of initial velocity and current velocity is positive. This requirement assures the traversing nature of the targets. Moreover, the changes of the targets' direction happen randomly within a time interval based on the properties of the targets, environment, and user's preferences. The sensors communicate among themselves when their state changes or when they detect the targets. A sensor detects evaders when they enter its circular sensing region (isotropic or omnidirectional sensors). The sensor collects data of the target's position when the target enters its sensing region for the first time. Hence, time instances of the detections are randomly distributed and cannot be anticipated. Since the sensors do not have perfect measurements, information regarding the position of targets are not accurate, but are prone to noise. As a consequence, a transition from one behavior of a sensor to another is based purely on stochastic events.

After $k-1$ independent detections of a target, a sensor is deployed to obtain the k^{th} independent observation of the target. We define a C-target as a region in the game area where the probability that a deployed sensor will get the k^{th} observation of the target is above a threshold ϵ determined by user's preferences (energy at disposal, percent of targets captured, etc.), and properties of both sensors and targets. C-targets can be approximated with cone-like areas as illustrated in Figure 2. In order to cover the C-target with a finite sensing region, the sensor moves orthogonally to the bisector of the C-target within the boundaries of the C-target. We require k independent detections before deploying a sensor into pursuit mode in order to avoid false alarms. The decision

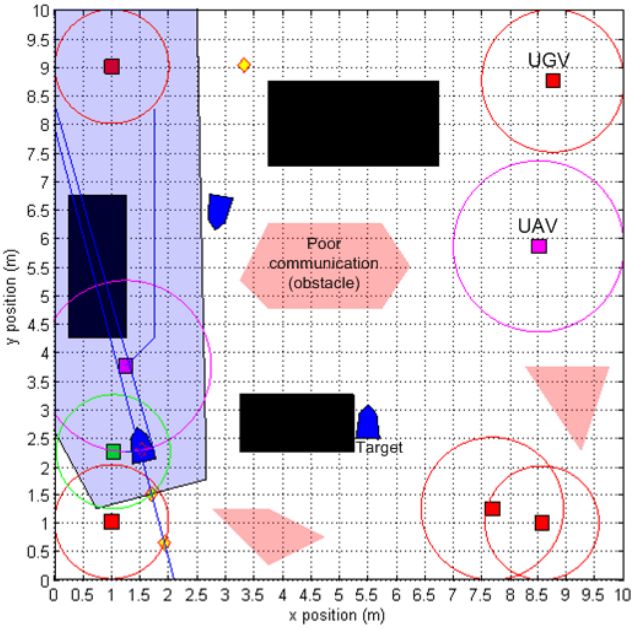

Fig. 2. An illustrative detail of a simulation. Red squares represent UGVs, the green square represents a UGV in pursuit mode, purple squares represent UGVs, circles denote corresponding sensing regions, blue polygons represent targets, black rectangles represent obstacles, and the blue shaded cone-like area is a C-target. Red shaded polygons represent areas with poor communication.

of which sensor to deploy for either of the above mentioned tasks is based on the reward function (defined in Section III). The function is designed for straight line moving targets in [10]. Therefore, the maximal value of detecting the targets cooperatively is obtained when the sensors are grouped in the corners of the rectangular game area (for a comprehensive discussion see [11]). To accommodate the optimality criteria for the targets in this paper, we maximize the reward function with the constraint that a certain minimal area coverage must be satisfied.

Next, in order to make the problem more realistic, we consider heterogeneous sensors (i.e., sensors with different properties) installed on robotic platforms with different functionalities. The sensors have different sensing regions while the platforms are UGVs and UAVs. Sensors on UGVs have smaller sensing areas than those on UAVs. In addition, UGVs are slower then UAVs, but UGVs can capture targets whereas UAVs cannot. Unlike ground vehicles, aerial vehicles can fly over obstacles. The properties of both sensors and platforms are taken into account for motion planning. Throughout the paper, we refer to both the sensors and associated platforms as sensors.

Finally, we assume that there are areas in the environment where communication is not possible or is very poor. We model these areas as virtual obstacles that sensors would avoid but targets can enter. Virtual obstacles due to communication are introduced in [12].

Uniform distribution is used throughout this paper because it

is 'the most random' distribution. Use of any other zero-mean distribution for modeling directions of the targets gives even better results in a sense of fulfilling the problem objectives.

III. Mathematical preliminaries and definitions

Mathematical preliminaries are divided into two subsections. In the first subsection, we present concepts and details of the hybrid model developed in this work. In the second subsection, the methods brought together in order to successfully solve the case study are conveyed. We combine cell decomposition, geometric optimization and track coverage into a cohesive framework.

A. Hybrid modeling

A stochastic hybrid model capable of describing a wide range of multi-agent problems is developed and applied to a multi-target tracking problem using heterogeneous mobile sensor networks.

Roughly speaking, hybrid systems are dynamical systems that involve the interaction of different types of dynamics (i.e., continuous state and discrete state dynamics). While modeling, discrete states are related to different modes of behavior such as sensing, pursuing or avoiding obstacles. These modes generally have different goals, continuous dynamics, control laws, sensing and communication policies.

A hybrid automaton is often used as a modeling language for hybrid systems. Merging discussions from [3] and [13], we define the following:

Definition 1: A hybrid automaton H is a tuple $H = (Q, X, f, \Upsilon, U, \Delta, D, Init, Dom, E, G, R)$ *that describes the evolution of*

- discrete state variables $q \in Q$ *and* continuous state variables $x \in X$,
- control inputs $\upsilon \in \Upsilon$ *and* $u \in U$, *and*
- stochastic or disturbance inputs $\delta \in \Delta$ *and* $d \in D$

by means of four functions:

- *a* vector field $f(\cdot,\cdot,\cdot,\cdot) : Q \times X \times U \times D \to X$,
- *a* domain set $Dom(\cdot,\cdot,\cdot) : Q \times \Upsilon \times \upsilon \to P(X)$,
- *a* guard sets $G(\cdot,\cdot,\cdot) : E \times \Upsilon \times \Delta \to P(X)$, *and*
- *a* reset function $R(\cdot,\cdot,\cdot,\cdot) : E \times X \times \Upsilon \times \Delta \to X$,

where $Init \subseteq Q \times X$ *is a* set of initial states *and* $E \subseteq Q \times Q$ *is a* set of edges.

In the above definition $P(X)$ denotes the power set of X. Furthermore, we refer to $(q, x) \in Q \times X$ as the state of H. We assume that the sets Q, Υ and Δ are countable and that $X = \mathbb{R}^n$ (or $\mathbb{C}^n, \mathbb{S}^n$), $U \subseteq \mathbb{R}^m$, and $D \subseteq \mathbb{R}^n$ for integers n, m, and p, where $\mathbb{C}^n$ is n-dimensional complex space, and $\mathbb{S}^n$ is n-dimensional sphere. It should be noted that function f is a mapping to TX (tangent space of X) with form

$$\dot{x}(t) = f(t, q_0, x_0, x(t), u(t), d(t)), t \in \mathbb{R}_{t\geq 0}, \qquad (1)$$

and is required to be continuous. With (1), the evolution $x : \mathbb{R}_{t\geq 0} \to X$ of the dynamical system starting at some initial state $(q_0, x_0) \in Init$, with input $u : \mathbb{R}_{t\geq 0} \to U$ and stochastic input $d : \mathbb{R}_{t\geq 0} \to D$ is given. The set of nonnegative real numbers is denoted with $\mathbb{R}_{t\geq 0}$.

To characterize the evolution of the state of a hybrid automaton, there is a need for an appropriate set of times. Such set has to capture both continuous intervals (over which continuous evolution takes place) and discrete points in time (when discrete transitions happen). This set of times is called a hybrid time set. From [13], we have,

Definition 2: A hybrid time set is a sequence of intervals $\tau = I_0, I_1, , I_N = \{I_i\}_{i=0}^{N}$, *finite or infinite (i.e.,* $N = \infty$ *is allowed) such that*

- $I_i = [\tau_i, \tau_i'], \forall i < N$;
- *if* $N < \infty$ *then either* $I_N = [\tau_N, \tau_N']$ *or* $I_N = [\tau_N, \tau_N')$
- $\tau_i \leq \tau_i' = \tau_{i+1}, \forall i$.

In multi-agent applications, each agent can be represented as a hybrid automaton. Such hybrid automata form a mobile hybrid network (MHN) of agents (e.g., networks of sensors and pursuers) that are able to interact within the network and with members of other networks. MHNs can be modeled such that each node represents a mobile agent (sensor or target) with communication, sensing, and control capabilities. Following some ideas from [1], we define a MHN as follows:

Definition 3: A mobile hybrid network Σ *is a tuple* $\Sigma = (\mathcal{I}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_s, \mathcal{G}_h)$, *where* $\mathcal{I} = \{1, \ldots, N\}$ *is the set of unique identifiers representing agents in the network,* $\mathcal{A} = \{H_i\}_{i\in\mathcal{I}}$ *is a set of control systems (physical agents) with processing power modeled as hybrid automata,* $\mathcal{G}_c = \{\mathcal{V}, \mathcal{E}_c\}$ *is a directed communication graph, where* $\mathcal{V}$ *is the set of nodes and* $\mathcal{E}_c$ *is the communication edge map, and* $\mathcal{G}_s = \{\mathcal{V}, \mathcal{E}_s\}$ *is a directed sensing graph, where* $\mathcal{E}_s$ *is the sensing edge map. Finally,* $\mathcal{G}_h = \{\mathcal{V}, \mathcal{E}_h\}$ *is a directed control graph with the set of nodes* $\mathcal{V}$ *and the control edge map* $\mathcal{E}_h$. *If* $H_i = H_j, \forall i, j \in \mathcal{I}$, *the network is uniform. Otherwise, the network is heterogeneous.*

Several graphs are needed to capture the interactions of the agents within the network and environment. In some cases, agents can 'hear' but not 'see' each other. We use proximity graphs to form $\mathcal{G}_c$ and $\mathcal{G}_s$. Proximity graphs provide a natural way to mathematically model the network interconnection topology resulting from the agents sensing and/or communication capabilities. Sensors exchange information using a communication protocol. The design of the control graph $\mathcal{G}_h$ involves the assignment of control policies for each agent. The set $\mathcal{E}_h$ is related to the communication $\mathcal{E}_c$ and sensing $\mathcal{E}_s$ graphs. An edge between two nodes in the control graph can be created only if a communication edge or sensing edge exists. Furthermore, processing capabilities of agents bring parallel processing (e.g., coverage optimization), hierarchical structure, (distributed) control and estimation into focus.

Based on the characteristics of the events (upon which state evolution of a hybrid automata forming the network depend) occurring in the network, a hybrid network could be synchronous, asynchronous or a combination of both. That bring us to the following definition:

Definition 4: A synchronous hybrid network is a set of hybrid systems where exists a scheduled increasing sequence of time instants $\mathbb{T} = \{t_k\}_{k\in\mathbb{N}}$ *or a sequence of events* $\mathbb{E} = \{e_k\}_{k\in\mathbb{N}}$ *that take place at* $\mathbb{T}_e = \{t_{e_k}\}_{k\in\mathbb{N}}$ *when executions of the hybrid automata happen.*

On the other hand, there is no such sequence in asynchronous networks. In the networks that are a combination of both, subsets of the network's elements are not mutually synchronized, but the elements within each subset are synchronized.

The evolution (or solution) of a hybrid automaton could be deterministic, nondeterministic and stochastic. The nondeterministic hybrid automata are those with certain freedom in defining a solution, while for given input and initial state of a deterministic hybrid automaton its state is uniquely defined at any instant of time in the future. Refining nondeterministic models in order to get better analysis of uncertain systems calls for a stochastic hybrid model. In such model, uncertainties (failures, duration of operations, switching between states, etc.) are modeled as random variables or random processes in order to include probabilistic phenomena.

B. Cooperative multi-target tracking - a case study

We consider a pursuit-evasion game where the set $\mathcal{P}$ of N heterogeneous robotic sensors have to detect, pursue and capture M randomly moving targets members of the set $\mathcal{T}$. Elements of set $\mathcal{P}$ and $\mathcal{T}$ are denoted $\mathcal{P}_i$ and $\mathcal{T}_i$ respectively. With $\mathcal{I}_\mathcal{P}$ we denote the index set of $\mathcal{P}$, and with $\mathcal{I}_\mathcal{T}$ the index set of $\mathcal{T}$. The game takes place in a polygonal area-of-interest $\mathcal{S} \subset \mathbb{R}^2$ with boundary $\partial\mathcal{S}$. The area $\mathcal{S}$ 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 $q_{\mathcal{P}_i}(t)$ that specifies its position and orientation at time t with respect to a fixed (or inertial) Cartesian frame $\mathcal{F}_\mathcal{S}$ related to $\mathcal{S}$. When dealing with targets, position and orientation of the i^{th} target at time t is comprised in $q_{\mathcal{T}_i}(t)$. Let us point out that $q_{\mathcal{P}_i}(t)$ and $q_{\mathcal{T}_i}(t)$ are 3-dimensional vectors, i.e., $q_{\mathcal{P}_i}(t) = [X_{\mathcal{P}_i}(t)\ Y_{\mathcal{P}_i}(t)\ \theta_{\mathcal{P}_i}(t)]^T$, and $q_{\mathcal{T}_i}(t) = [X_{\mathcal{T}_i}(t)\ Y_{\mathcal{T}_i}(t)\ \theta_{\mathcal{T}_i}(t)]^T$. We use $\theta_{\mathcal{T}_i}(t)$ interchangeably with $\theta_{\mathcal{T}_i,t}$ throughout the paper.

At this stage of the implementation, the orientation of pursuers does not play a significant role since sensors are assumed omnidirectional and holonomic. Therefore, we have,

$$\dot{q}_{\mathcal{P}_i}(t) = u_i(t), \tag{2}$$

where $u_i(t) \in \mathbb{R}^2$ is the input of the pursuer $\mathcal{P}_i$.

Let us introduce the following definitions in order to proceed. From [14] we have,

Definition 5: A continuous-time random process is a family of random variables X_t where t ranges over a specified interval of time.

Definition 6: We say that X_t is a continuous-time Markov process if for $0 \leq s_0 < \cdots < s_{n-1} < s < t$ we have $\Pr\{X_t \in B | X_s = x, X_{s_{n-1}} = x_{n-1}, \cdots, X_{s_0} = x_0\} = \Pr\{X_t \in B | X_s = x\}$ *where* Pr *denotes probability.*

As a consequence of Definition 5, a target $\mathcal{T}_i$ is a continuous-time random process of random variables $\theta_{\mathcal{T}_i,t}$ where t ranges from t_0 to ∞ (t_0 is time when target entered $\mathcal{S}$). The random variable $\theta_{\mathcal{T}_i,t}$ describes the target's direction. A three-dimensional real valued vector function (components are continuous functions) maps the family of random variables $\theta_{\mathcal{T}_i,t}$ into $q_{\mathcal{T}_i}(t)$. At particular time instant t, the value of process $\mathcal{T}_i$ is given by $q_{\mathcal{T}_i}(t)$. Notice that the future value of $q_{\mathcal{T}_i}(t)$ at any given time is a function of target's current velocity, current position and $\theta_{\mathcal{T}_i,t}$ given with the following expressions,

$$\begin{aligned} \dot{X}_{\mathcal{T}_i}(t) &= v_{\mathcal{T}_i}(t)\cos\theta_{\mathcal{T}_i}(t), \\ \dot{Y}_{\mathcal{T}_i}(t) &= v_{\mathcal{T}_i}(t)\sin\theta_{\mathcal{T}_i}(t), \end{aligned} \tag{3}$$

and therefore, $\mathcal{T}_i$ is a Markov process. The third component of the vector function is the identity function.

The first two components of $\mathcal{T}_i$ are piece-wise linear, while the third component has discontinuities at time instants when a change in direction happens. The implemented time span between two consecutive instants is uniformly distributed in interval $[T_{i,\min}, T_{i,\max}]$ determined by the user's preferences. A change of direction is uniformly distributed in interval $(\theta_{\mathcal{T}_i}(t_0) - \pi/2,\ \theta_{\mathcal{T}_i}(t_0) + \pi/2)$. Therefore, $\theta_{\mathcal{T}_i,t}$ is uniformly distributed random variable for countably many time instants t (i.e., a sequence of independent identically distributed random variables), and otherwise is deterministic (holds a value of the last change of the target's direction).

The maximum translational speed of all sensors and targets is known, and $v_{\mathcal{P},\max} > v_{\mathcal{T},\max}$. While sensors can move with any speed in $[0, v_{\mathcal{P},\max}]$, it is assumed that the speed of every target is uniformly distributed in $[v_{\mathcal{T},\min}, v_{\mathcal{T},\max}]$, with $v_{\mathcal{T},\min} > 0$.

Let $\mathcal{C}_{free,i}$ denote the configuration space of the i^{th} sensor that is free of obstacles and other sensors. Let $\mathcal{F}_{\mathcal{A}_i}$ denote a moving Cartesian frame embedded in $\mathcal{A}_i$. If we assume that $\mathcal{D}_i$ (the geometry of the sensor's field-of-view) and $\mathcal{A}_i$ are both rigid, then $q_{\mathcal{P}_i}(t)$ also specifies the position of every point in $\mathcal{D}_i$ (or $\mathcal{A}_i$) relative to $\mathcal{F}_\mathcal{S}$. Using the $k-1$ individual sensors' detections up to time τ where $k \in \mathbb{N}$, it is possible to identify the area in $\mathcal{S}$ where the sensor may obtain measurements of a target with the probability higher then some threshold. That leads to the following definition and proposition.

Definition 7: The target $\mathcal{T}_j$ in $\mathcal{S}$ maps in the i^{th} sensor configuration space $\mathcal{C}$ to the C-target region $\mathcal{CR}_j = \{q_i \in \mathcal{C} \mid \Pr\{\mathcal{D}_i \cap \mathcal{T}_i\} > \epsilon, \forall t \geq \tau, i \in \mathcal{I}_\mathcal{P}, j \in \mathcal{I}_\mathcal{T}\}$.

Proposition 1: C-target can be approximated with cone-like area (as shown in Figure 2).

Proof: Let $\mathcal{F}_i$ be a Cartesian coordinate system associated to C-target of target i. Let its y-axis be a minimum squared error line with respect to $k-1$ detection points of the target. Note that the y-axis is a bisector of the C-target. The x-axis contains a point of the $k-1^{th}$ detection. For the sake of simplicity, let us assume that the change of direction happens periodically so the distance covered by the target in one period is d. Let us define sums of independent random variables $A_n = \sum_{i=1}^{n} d\cos\theta_i$ and $B_n = \sum_{i=1}^{n} d\sin\theta_i$. Expectation of A_n is 0 and of B_n is $\frac{2nd}{\pi}$ while variances are $\frac{nd}{2}$ and $n\frac{d\pi^2 - 8d^2}{2\pi^2}$, respectively. Therefore, as n grows, the likelihood that the target could be found further away from the bisector increases. Moreover, the target is more likely to progress along the bisector. Since $\frac{B_n - B_{n-1}}{A_n - A_{n-1}} = \tan\theta_n$, using $\Pr\{0 \leq \tan\theta_n \leq \alpha\} = \frac{1-\epsilon}{2}$ we can approximate the

boundaries of the C-target with lines. Their slopes are α and $\pi - \alpha$ with respect to $\mathcal{F}_i$ where $\alpha = \tan(\pi \frac{1-\epsilon}{2})$. Intersections of the lines with x-axis of $\mathcal{F}_i$ are given with $\pm r_i \epsilon$ where r_i is the radius of the sensor. ■

A target with $k-1$ independent detections is called a partially-observed target. A sensor receiving the highest reward is deployed to investigate the C-target of the partially-observed target making maneuvers described in Section II. After gathering k independent observations of the target $\mathcal{T}_i$, a sensor with highest reward is deployed to capture the target. Our goal is to estimate the future position of the target and use a pursuit strategy that maximizes the likelihood of capturing the target. Therefore, based on k intermittent observations of the target's position, the following capturing policy is proposed and implemented: *Move to the point of the last detection. Afterwards, move to the intersection of minimum squared error line and* $\partial\mathcal{S}$. The error that is minimized is the sum of squared distances of k detection points and the line.

In order to reduce computational complexity of uncountable space $\mathbb{R}^2$, a cell decomposition of $\mathcal{S}$ is implemented. We discretized the environment using uniform cell decomposition. Uniform cell decomposition is used because of its implementation tractability and optimal dispersion (see [4]). Optimal dispersion (in $\mathcal{L}_2$ norm) is important since the sensing regions are circles with finite area. Based on the cell decomposition, a connectivity graph $\mathcal{G}$ is obtained. Every sensor has its own connectivity graph. Cells forming $\mathcal{C}_{free,i}$ are divided into void and observation cells. Void cells are cells with the probability of detection of partially-observed targets less than threshold ϵ, while observation cells are those with the probability larger than the threshold. The connectivity graph, void and observation cells of each sensor have to be updated as the game progresses. Cells in the decomposition (nodes of the graph) are denoted k_i, and sensors (except in pursue mode and initial placement) move among centroids of the cells. Therefore, we have a framework to define the underlying performance index in order to achieve the goals stated in Section II.

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 $q_{\mathcal{T}_i}(t_1) \in k_l$ to a configuration in an adjacent cell $q_{\mathcal{T}_i}(t_2) \in k_i$ (obviously, $t_1 < t_2$) taking into account distance. The reward function is as follows:

$$R(k_l, k_i) = P_{\mathcal{R}}(k_i) + \Delta P^k_{\mathcal{S}}(k_l, k_i) - d(k_l, k_i), \qquad (4)$$

where $P_{\mathcal{R}}$ is the probability of cooperatively detecting a partially-observed target, $\Delta P^k_{\mathcal{S}}$ is the gain in the probability of cooperatively detecting unobserved targets and $d(k_l, k_i)$ is distance between centroids of the cells. An unobserved target is a target with less then $k-1$ independent observations. These probability density functions are obtained using the methodology based on geometric properties of sensors and area-of-interests described in [10]. The performance index (4) is maximized with the constraint that minimal area coverage has to be satisfied because of the reason stated in Section II. The reason is that the maximal value of $P^k_{\mathcal{S}}$, contributing as difference in (4), is obtained when the sensors are grouped in the corners of $\mathcal{S}$. Based on the reward function, a sensor with maximal reward along a path in the graph is deployed. The terms in the reward function are weighted based on user's preferences. While choosing a sensor for pursuit, more weight is put on the distance term since the pursuit is a costly operation. In order to find a sensor $\mathcal{P}_i$ with maximal path reward, we use graph searching algorithm A*. After finding an optimal sensor, we determine a control input $u_i(t)$ in (2) corresponding to the optimal path.

IV. Hybrid system model

The sensors (or pursuers) form MHN $\Sigma_{\mathcal{P}}$, and the targets (or evaders) form MHN $\Sigma_{\mathcal{T}}$. The sensors are fully connected, and the associated control graph is omitted since the sensors do not perform a coordinated motion as a group (e.g., keeping some formation) to accomplish the goal. Through communication, sensors exchange their current position. Therefore, a sensing graph is redundant, but sensing capability is essential for target tracking. The sensors are able to sense targets (i.e., a sensing between nodes of different networks takes place). Each new observation triggers exchange of the information between the sensors causing a change of the behavior, i.e., network $\Sigma_{\mathcal{P}}$ is synchronized by events related to targets. On the other hand, collision avoidance with other sensors gives $\Sigma_{\mathcal{P}}$ asynchronous behavior.

When considering sensors as processing units, the estimation of the targets' position is distributed among sensors, so as sensors' motion planing algorithms. In other words, multiple sensors are estimating the random processes (targets positions) joining information they have collected, while the control policy for each sensor is obtained considering only its configuration space. Each sensor calculates its reward function given in (4) communicating it to the others. The ground sensors' modes are:

- *sensing* (*static* or *mobile* with *avoiding obstacles*),
- *pursuing* (with *obstacle avoidance*), and
- *communicating* and *updating information*.

Aerial sensors do not have *pursuing* mode. Collisions among the sensors are avoided using model prediction. Since the motion planing of the sensors is distributed among them, such collisions are possible. Using one step look ahead, collisions are avoided switching to *avoiding obstacles* mode.

On the other hand, in this case study we assume that targets cannot communicate within their network, and each target is independent from other targets. Hence, network $\Sigma_{\mathcal{T}}$ is purely asynchronous. Evolution of $\Sigma_{\mathcal{T}}$ is stochastically modeled as described in Section III. They are only capable of sensing the obstacles in their vicinity. Target modes are:

- *active* (*avoiding obstacles* or *changing direction*), and
- *captured*.

Collision avoidance among the targets and obstacles is obtained using model prediction. Targets look one step ahead, and if there is a collision, they switch to the *avoiding obstacles* mode.

```
Perform initial optimal sensor placement
Decompose environment into C_free and C_obstacle cells
for all Sensors do
    Calculate obstacle and coverage maps
end for
while Game not over do
    for all Targets do
        if Time for change of direction then
            Change direction
        end if
        Update position (avoid obstacles, collisions and add noise)
    end for
    for all Sensors do
        if Sensor update interval then
            Calculate obstacle and coverage maps
        end if
        Update position (do maneuver, avoid obstacles and collisions)
        if Position changed then
            Update sensors' network information
        end if
        Detect targets
        if A target beneath capture threshold then
            Remove target
            Update sensors' network information
            End associated pursuit or investigations
        end if
    end for
    if Pursued target beneath capture threshold then
        Remove target
        End pursuit
        Update sensors' network information
    end if
    if Detection then
        Update sensors' network information
        if Target detections = k − 1 then
            Hypothesize target track
            Calculate observation cells
            for all Sensors that have not detected this target do
                Calculate path and reward to investigate target
            end for
            Deploy a sensor with the greatest reward
            Determine maneuver
        else if Target detections = k then
            for all Sensors not in pursuit do
                Calculate path and reward to pursue target
            end for
            Deploy a sensor with the greatest reward
        end if
    end if
end while
```

Fig. 3. Algorithm developed for the simulation.

The algorithm in Figure 3 clearly shows relations between aforementioned modes and events triggering transitions from one mode to the other.

V. Simulations and numerical results

The information-driven sensor planning and pursuit strategies described in previous sections are integrated in a simulator. A pseudo-code of the implemented algorithm is shown in Figure 3. We use $k = 3$ for the reasons stated in [15]. Due to space limitations of the paper, we omit the most of this section.

The robustness of the implemented strategies is verified by introducing noise in the targets' position and velocity.

VI. Conclusion

This paper presents a comprehensive stochastic hybrid model of mobile agent networks able to capture a wide range of multi-agent phenomena. A hybrid network consists of mobile agents modeled as hybrid systems with processing capabilities. The versatility and the flexibility of the model are demonstrated by the cooperative multi-target tracking case study fulfilling demanding tracking and pursuit goals. In the future, we plan to analyze the complexity of the approach applied in the case study, and to develop strategies involving more intelligent targets. In addition, we will apply the model to more progressive scenarios showing its full potential.

Acknowledgment

This work 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 was supported in part by the Office of Naval Research (Code 321), and by NSF grant ECS CAREER #0448906.

References

[1] F. Bullo, J. Cortes, and S. Martinez, *Distributed Control of Robotic Networks*. online book, 2008.

[2] M. Zavlanos and G. Pappas, "Distributed hybrid control for multiple pursuer multiple evader games," in *10th International Conference on Hybrid Systems: Computation and Control*, Pisa, Italy, April 2007, pp. 787–789.

[3] C. Cassandras and J. Lygeros, *Stochastic Hybrid Systems*. CRC Press, 2006.

[4] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[5] S. Ge and F. Lewis, Eds., *Autonomous Mobile Robots: Sensing, Control, Decision-Making, and Applications*. Boca Raton, FL: CRC Press, Taylor and Francis Group, 2006, chapters 10., 11. and 13.

[6] R. Sanfelice and E. Frazzoli, "A hybrid control framework for robust maneuver-based motion planning," in *American Control Conference*, Seattle, WA, July 2008, pp. 2254–2259.

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

[8] 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

[9] 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.

[10] S. Ferrari, R. Fierro, B. Perteet, C. Cai, and K. Baumgartner, "A geometric optimization approach to detecting and intercepting dynamic targets using a mobile sensor network," *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 292–320, 2009.

[11] K. Baumgartner, "Control and optimization of track coverage in underwater sensor networks," Ph.D. dissertation, Duke University, North Carolina, December 2007.

[12] A. Ghaffarkhah and Y. Mostofi, "Communication-aware target tracking using navigation functions - centralized case," in *Int. Conf. on Robot Communication and Coordination (RoboComm)*, Odense, Denmark, April 2009.

[13] J. Lygeros, "Lecture notes on hybrid systems," Department of Electrical and Computer Engineering, University of Patras, Greece, 2004.

[14] J. Gubner, *Probability and Random Processes for Electrical and Computer Engineers*. Cambridge University Press, 2006.

[15] T. Wettergren, R. Streit, and J. 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.