49th IEEE Conference on Decision and Control
December 15-17, 2010
Hilton Atlanta Hotel, Atlanta, GA, USA

A Randomized Hybrid System Approach to Coordinated Robotic Sensor Planning

Wenjie Lu, Guoxian Zhang and Silvia Ferrari

Abstract— **This paper proposed a randomized hybrid system approach for planning the paths and measurements of a network of robotic sensors deployed for searching and classifying objects in a partially-observed environment containing multiple obstacles and multiple targets. The sensor planning problem considered in this paper consists of coordinating and planning the motions of each robot, equipped with two on-board sensing capabilities. One sensing capability is assumed to have low classification accuracy and a large field of view (FOV), and the other is assumed to have high classification accuracy and a smaller FOV. A sampling function for rapidly-exploring trees is presented that takes into account both sensor measurements of obstacle locations and robot configuration and velocity to generate new milestones for the tree online. The tree expansion also takes into account the expected information value of the targets, represented by conditional mutual information, in order to favor expansions toward targets with higher measurement benefit. The proposed method is implemented and demonstrated on a network of robotic sensors simulated using the 3D physics-based robotics software packages Player/Stage/Gazebo.**

I. INTRODUCTION

Recently, it has been shown by several authors that the geometry of sensor's field-of-view (FOV), or visibility region, plays an important role in sensor motion planning [1]–[8]. The FOV can be considered as the region in which the sensor can make measurements. Typically, it consists of a bounded subset of a Euclidian space that depends on the sensor's configuration (e.g., position and orientation), and the sensor's parameters. Therefore, when a sensor is installed on a mobile robot, such as an unmanned ground vehicle, its configuration and FOV geometry must be taken into account in planning the robotic sensor path [5]–[8]. In many applications, such as, robotic mine hunting [9], cleaning [10], and monitoring of urban driving [11], industrial plants [12] or endangered species [13], multiple, cooperating robotic sensors are deployed in an obstacle-populated environment. Therefore, the geometries and configurations of the robot and the obstacles must also be taken into account to avoid collisions with obstacles and other robots [10], [14], [15], while minimizing the distance traveled, time, or fuel consumption.

Traditional path planning approaches, such as cell decomposition and potential field methods, have been successfully modified to address sensor path-planning problems for a single robotic sensor in a two-dimensional workspace with known obstacle geometries and locations [5], [8], [15]. This paper considers the problem of planning the paths of multiple robotic sensors operating in a partially-unknown three-dimensional workspace. Randomized path planning methods, such as probabilistic roadmap (PRM) and rapidly-exploring random trees (RRT), are known to compute solutions efficiently in high-dimensional problems [7], [16], [17].

RRT methods expand a trivial tree, containing the robot's initial configuration, iteratively over time, by performing two key steps during each iteration. In the first step, a random configuration is sampled from a known probability density function, and in the second step it is connected to the nearest node in the existing tree by an arc, with a predetermined distance. If the path between the two configurations is collision free, a new configuration and arc are added to the existing tree and, otherwise, they are discarded and a new random configuration is re-sampled. RRT methods have been successfully applied to many path planning problems. A method known as RRT-Connect was used in [18] to solve a path planning problem without the need to constrain the search to a pre-defined distance. In [19] a method known as CL-RRT was proposed for path planning in real-time autonomous urban driving. CL-RRT first selects the best node in current tree to connect a new sample configuration, and then evaluate its feasibility by generating a trajectory which lies in free configuration space and satisfies all other navigation constraints.

In this paper, we modify the RRT algorithm proposed in [19] to plan the paths of multiple robotic sensors based on their sensing objectives, namely target classification, and on the geometry of their FOVs. This modified RRT method is combined with hybrid system theory, to obtain a so-called randomized hybrid system approach for cooperatively planning and coordinating the motion of multiple robotic sensors that are deployed to classify multiple targets in a common workspace through sensor fusion. In formation control, the control of each robot is affected by other members as well as the group objective. Once new targets are detected by a sensor, they are assigned to a sensor in the network based on a tradeoff between distance and expected information benefit, such that classification may be improved through sensor fusion over time. The approach is demonstrated by using the robotics software packages is the Player/Stage/Gazebo (PSG) [20]. The PSG project consists of libraries that provide access to communication and interface functionality on robot hardware. Users write client applications, such as control algorithms, that connect to and command modules robot drivers running on a Player server, and are then able to visualize the results using a 3D physics-based simulation

Wenjie Lu, Guoxian Zhang and Silvia Ferrari are with the Laboratory for Intelligent Systems and Control (LISC), Department of Mechanical Engineering, Duke University, Durham, NC 27708-0005, {wenjie.lu, guoxian.zhang, sferrari}@duke.edu

environment called Gazebo.

This paper is organized as follows. Section II describes the sensor planning problem formulation and assumptions. The background on RRT method is reviewed in section III. Section IV describes the novel randomized hybrid system approach presented in this paper, and the implementation and simulation results are shown in section V and VI, respectively.

II. PROBLEM FORMULATION AND ASSUMPTIONS

This paper addresses the problem of planning the paths of a network of r robotic sensors with the same platform geometry $\mathcal{A}_i = \mathcal{A} \in \Re^3$, $i = 1, 2, \ldots, r$, a so-called accurate FOV, denoted by $\mathcal{S}_i = \mathcal{S} \in \Re^3$, $i = 1, 2, \ldots, r$, and an approximate FOV, denoted by $\mathcal{G}_i = \mathcal{G} \in \Re^3$, $i = 1, 2, \ldots, r$, all of which are assumed to be connected compact subsets of $\Re^3$. Let I_A denote the index set of the robotic platforms. Every sensor is assumed to be fixed on the robot platform, and to navigate a common partially-observed workspace denoted by $\mathcal{W}$ to measure and classify multiple geometric targets. Prior information, such as airborne sensor measurements and environmental maps, is used to estimate the geometry and location of obstacles and targets in B and T. However, we assume only a portion of target and obstacle locations are known *a priori*. In this paper, $\mathcal{A}$, $\mathcal{S}$ and $\mathcal{G}$ are modeled as convex objects. The workspace is assumed to be a compact (i.e., closed and bounded) subset of a three-dimensional Euclidian space, $\mathcal{W} \in \Re^3$, and to be populated with n fixed obstacles $B = \{\mathcal{B}_1, \ldots, \mathcal{B}_n\}$ and m fixed targets $T = \{\mathcal{T}_1, \ldots, \mathcal{T}_m\}$ with $\mathcal{B}_i \cap \mathcal{T}_j = \emptyset$, $\forall i \in I_B$ and $j \in I_T$, where I_B and I_T are the index sets of obstacles and targets. An example of the workspace in two dimensional space is shown in Fig. 1.

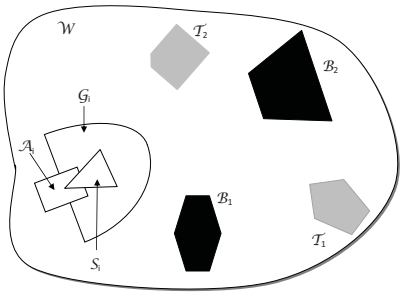

Fig. 1. Relevant problem geometries and notation.

Let $\mathcal{F}_{\mathcal{A}i}$ be a moving Cartesian frame embedded in $\mathcal{A}_i$. Then, every point of $\mathcal{S}_i$, and every point of $\mathcal{G}_i$ have a fixed position with respect to $\mathcal{F}_{\mathcal{A}i}$, and the configuration $\mathbf{q}_i = (x_i \ y_i \ \theta_i) \in SE(2)$ is used to specify the position (x_i, y_i) and orientation θ_i of all $\mathcal{A}_i$, $\mathcal{S}_i$ and $\mathcal{G}_i$ with respect to a fixed inertial frame $\mathcal{F}_{\mathcal{W}}$, embedded in $\mathcal{W}$. Obstacles and targets are also assumed to be fixed and rigid in $\mathcal{W}$, such that every point of $\mathcal{B}_i$, for $\forall i \in I_B$, and every point of $\mathcal{T}_j$, $\forall j \in I_T$, have a fixed position with respect to $\mathcal{F}_{\mathcal{W}}$. Let $\mathcal{C}$ denote the space of all possible robot configurations. Then, the path of the ith robotic platform's centroid is defined as a continuous map $\tau_i : [0, 1] \to \mathcal{C}$, with $\mathbf{q}_{i_0} = \tau_i(0)$ and $\mathbf{q}_{i_f} = \tau_i(1)$, where $\mathbf{q}_{i_0}$ and $\mathbf{q}_{i_f}$ are the initial and final configurations, respectively. Since $\mathcal{S}_i$ and $\mathcal{G}_i$ are mounted on $\mathcal{A}_i$, the path τ_i determines the targets in $\mathcal{W}$ that can be measured by this robotic sensor, while traveling from $\mathbf{q}_{i_0}$ to $\mathbf{q}_{i_f}$. Let $Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_r\}$ be the set of robotic sensors' configurations, and $Q_0 = \{\mathbf{q}_{1_0}, \ldots, \mathbf{q}_{r_0}\}$ and $Q_f = \{\mathbf{q}_{1_f}, \ldots, \mathbf{q}_{r_f}\}$ denote the sets of initial and final sensors' configurations, respectively. Then, the set of paths $\Gamma = \{\tau_1, \ldots, \tau_r\}$ determines the targets in $\mathcal{W}$ that can be detected and classified by the robotic sensor network traveling from Q_0 to Q_f.

It is assumed that the measurement process of every sensor in the network can be modeled by a known joint probability mass function (PMF) obtained from first principles or prior experiments [5], [7], [21], [22]. Let $\mathbf{Z}_i \in \mathcal{Z} \subset \Re^r$ denote the sensor measurements from target $\mathcal{T}_i \in T$ that are used to estimate or classify the unknown target state $\mathbf{X}_i \in \mathcal{X} \subset \Re^n$. The sensor characteristics, including the sensor mode, environmental conditions, and sensor noise or measurement errors, are grouped in a vector of parameters $\boldsymbol{\lambda}_i \in \Re^l$. Then, assuming that the targets' state, sensor measurements, and parameters are random vectors, the sensor measurements can be modeled by a joint PMF that typically can be factorized as follows [5], [7], [23], [24],

$$p(\mathbf{Z}_i, \mathbf{X}_i, \boldsymbol{\lambda}_i) = p(\mathbf{Z}_i \mid \mathbf{X}_i, \boldsymbol{\lambda}_i) p(\mathbf{X}_i) p(\boldsymbol{\lambda}_i), \ \forall i \in I_T \quad (1)$$

assuming that $\mathbf{X}_i$ and $\boldsymbol{\lambda}_i$ are independent variables. In this paper, the sensor model represented by (1) is considered to hold for all targets, and to remain constant regardless of measuring distance at all times. A convenient approach for modeling the sensor PMF in (1) is to construct a Bayesian network(BN) from prior sensor data and experiments [24].

The robotic sensor network is deployed to search and classify targets in $\mathcal{W}$, based on partial prior information about the targets' and obstacles' locations and geometries. Therefore, the path planning algorithm must take into account both the value of exploration, for detecting new targets, and the value of information, for correctly classifying targets that have been detected up to the present time by airborne or ground robotic sensors. Once a target $i \in I_T$ is detected, we assume that its location and geometry $\mathcal{T}_i$ become known, but its classification $\mathbf{X}_i$ remains uncertain, due to the random nature of the measurement process (1). Therefore, the ith target's information value, denoted by V_i, is the expected benefit of making additional measurements from $\mathcal{T}_i$ and, as shown in Section IV, it can be represented by the expected reduction in uncertainty associated with $\mathbf{Z}_i$, conditioned on the sensor model and prior information.

Since each sensor is assumed fixed and mounted on a robotic platform, $\mathcal{S}_i$ and $\mathcal{G}_i$ are fixed with respect to $\mathcal{A}_i$, the platform motions must be planned in concert with the sensor exploration and measurement processes. Let the measurement set of the r robotic sensors along their paths $\tau_1, \ldots, \tau_r$ be defined as $Z = \{\mathbf{Z}_j \mid \mathcal{T}_j \cap \mathcal{S}_i(\mathbf{q}_i) \neq \emptyset, \ \tau_i(s) = \mathbf{q}_i, \ s \in [0, 1], \ \forall j \in I_T, \ \forall i \in I_A\}$, where $\mathcal{S}_i(\mathbf{q}_i)$ is the

subset of $\mathcal{W}$ occupied by $\mathcal{S}_i$ at a configuration $\mathbf{q}_i$, along τ_i. Then, the robotic sensors paths must achieve the following objectives: (i) explore $\mathcal{W}$ to detect new targets with $\mathcal{G}_i$, (ii) maximize the information value of Z, (iii) avoid all obstacles and (iv) robots in $\mathcal{W}$, while traveling the minimum total distance between Q_0 and Q_f. Thus, while all platforms A must avoid intersections (collisions) with each other and with the obstacles B, the accurate FOVs G must explore $\mathcal{W}$, and the accurate FOVs S must intersect $\mathcal{T}_j$, and obtain additional measurements $\mathbf{Z}_j$ for classification.

III. Background on RRT Path Planning

Rapidly-Exploring Random Trees (RRTs), introduced by Lavalle in [25], provide an efficient way to search for a path in a configuration space online, and have been successfully applied to nonholonomic robots in high-dimensional workspaces. Using the initial robot configuration $\mathbf{q}_{i_0}$, at time $t_k = 0$ the initial tree is defined as $Tr(t_k) = \mathbf{q}_{i_0}$, and is expanded as follows, by iterating incrementally over the discrete time index $t_k = 1, 2, \ldots$. First a configuration $\mathbf{q}$ is randomly sampled in $\mathcal{C}_{free}$ using a PDF $p(\mathbf{q})$. Then, based on a distance metric, the closest node to $\mathbf{q}$ in $Tr(t_k)$ is computed, and extended toward $\mathbf{q}$ within a predefined distance ϵ and obtains $\mathbf{q}'$. If the path lies in $\mathcal{C}_{free}$, $\mathbf{q}'$ is added to $Tr(t_k)$, otherwise, it is discarded.

In [18], a modified RRT method was proposed for extending the nearest node in the current tree to the sample unless an obstacle is reached. Another extension of RRT is to bias the sample distribution based on a reference configuration (x, y, θ), using the following equation,

$$\begin{pmatrix} x^s \\ y^s \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} cos(\theta^s) \\ sin(\theta^s) \end{pmatrix} (|l^s| + l_0) \tag{2}$$

$$\begin{aligned} \theta^s &\sim N(\theta, \sigma_1^2) \\ l^s &\sim N(0, \sigma_2^2) \end{aligned} \tag{3}$$

taken from [19], where (x^s, y^s, θ^s) is the sampled configuration, and $N(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2. In this paper, the vehicle path planning method presented in [19] is modified for planning the paths of a cooperative sensor network, by introducing a sampling method in which the PDF $p(\mathbf{q})$ is generated based on the geometry and information value of the C-obstacles $\mathcal{CB}$, using a normal mixture. The sampled configurations are ordered based on the robotic sensor state, the expected information value of the target assigned to the sensor, and the distance to the target, as explained in the next section.

IV. Methodology

In this section, a hybrid system approach is proposed for coordinating a robotic sensor network deployed to detect and classify multiple targets in a partially-observed workspace. A modified RRT approach is presented to navigate the workspace in search for important targets while avoiding collisions with obstacles. A corresponding potential navigation function is designed to avoid collisions between robotic sensors.

A. Hybrid Model of Robotic Sensor Network

Assume that the network of robotic sensor explores the environment simultaneously and can communicate and share the workspace information. A hybrid system model of sensors, targets, and obstacles is developed and illustrated in Fig. 2. The state of the system is updated when one of the following events takes place: a new target $\mathcal{T}_i$ is detected by a sensor FOV $\mathcal{G}_k$, i.e., $\mathcal{G}_k \cap \mathcal{T}_i \neq \emptyset$; a new obstacle $\mathcal{B}_j$ is detected by a sensor FOV $\mathcal{G}_k$, i.e., $\mathcal{G}_k \cap \mathcal{B}_i \neq \emptyset$; or, a target is measured by a high accuracy sensor $\mathcal{S}_i$. When a new target $\mathcal{T}_i$ is detected, the measurement $\mathbf{Z}_i$, is used to compute the expected information value and plan future measurements from the same target. With every event, the detected targets are assigned to the nearest robotic sensor, and the corresponding potential field is updated based on the current information. The dynamics of each robotic platform are approximated by a unicycle model given by,

$$\begin{pmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{pmatrix} = \begin{pmatrix} cos(\theta_i) & 0 \\ sin(\theta_i) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v_i \\ w_i \end{pmatrix} \tag{4}$$

where v_i is the linear velocity, w_i is the angular velocity, The linear velocity command v_i^c, and the angular velocity command w_i^c are used as the reference for a lower-level feedback controller to track the reference trajectory τ_i.

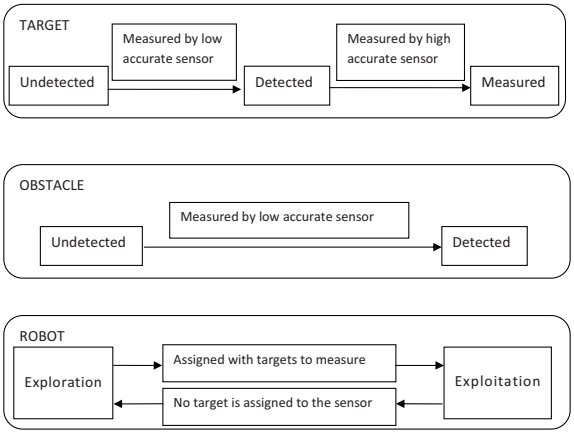

Fig. 2. Finite-state model of robotic sensors, targets, and obstacles.

To avoid collisions with each other, the robotic sensors are coordinated according to the following rules. Sensors in the exploration state are assumed to have lower priority than sensors in the exploitation state. For the jth sensor at configuration $\mathbf{q}$, any other sensor in $\mathcal{W}$ is considered as an obstacle. Therefore, the corresponding C-obstacle of the ith sensor, denoted by $\mathcal{CA}_i$, can be generated from the two robots' geometries as explained in [26]. A repulsive potential function can then be used to avoid collisions between a sensor with configuration $\mathbf{q}$ and the ith sensor. The repulsive

potential is defined as,

$$U_i(\mathbf{q}) = \begin{cases} \frac{1}{2}\eta_1(\frac{1}{\rho(\mathbf{q},\mathbf{q}_i)} - \frac{1}{\rho_0})^2 & \text{if} \quad \rho(\mathbf{q},\mathbf{q}_i) \leq \rho_0 \\ 0 & \text{if} \quad \rho(\mathbf{q},\mathbf{q}_i) > \rho_0 \end{cases} \tag{5}$$

where η_1 is a scaling parameter, ρ_0 is the influence distance of each agent, $\mathbf{q}_i$ is the ith sensor's configuration, and $\rho(\mathbf{q},\mathbf{q}_i)$ is the shortest distance between $\mathbf{q}$ and $\mathbf{q}_c$ for all $\mathbf{q}_c \in \mathcal{CA}_i$. The gradient of the repulsive potential, given by,

$$\nabla U_i(\mathbf{q}) = \begin{cases} \eta_1(\frac{1}{\rho(\mathbf{q},\mathbf{q}_i)} - \frac{1}{\rho_0})\frac{\nabla\rho(\mathbf{q})}{\rho(\mathbf{q},\mathbf{q}_i)^2} & \text{if } \rho(\mathbf{q},\mathbf{q}_i) \leq \rho_0 \\ 0 & \text{if } \rho(\mathbf{q},\mathbf{q}_i) > \rho_0 \end{cases} \tag{6}$$

can then be added to the lower-level feedback controller to track τ_i, while avoiding $\mathcal{A}_i$.

At $t_k = 0$, the robotic sensors are randomly placed in the workspace $\mathcal{W}$, in the exploration state. At every time step t_k, the index sets $I_t(t_k)$ and $I_r(t_k)$ represent the indices of robotic sensors in exploitation and exploration states, respectively. Suppose the ith robotic sensor is in the exploration state, at configuration $\mathbf{q}_i$. Then, the set of its neighbors is defined as $N_r(\mathbf{q}_i) = \{j \mid \rho(\mathbf{q}_i,\mathbf{q}_j) < \rho_0,\ \forall j \in I_A, j \neq i\}$. When $N_r(\mathbf{q}_i) \neq \emptyset$, all known obstacles are also taken into account to generate the repulsive potential for the ith sensor, and is computed similarly to (5). Therefore, the artificial potential field of the ith robotic sensor is,

$$U_r(\mathbf{q}_i) = \sum_{j\in N_r(\mathbf{q}_i)} U_j(\mathbf{q}_i) + \sum_{j\in I_B} U_j(\mathbf{q}_i) \tag{7}$$

and the artificial force applied to the ith robotic sensor by the feedback controller is,

$$F_r(\mathbf{q}_i) = -\nabla U_r(\mathbf{q}_i) = -\sum_{j\in N_r(\mathbf{q}_i)} \nabla U_j(\mathbf{q}_i) - \sum_{j\in I_B} \nabla U_j(\mathbf{q}_i) \tag{8}$$

The angular velocity command is,

$$w_i^c = \begin{cases} \pm\frac{\angle F_r(\mathbf{q}_i) - \theta_i}{\Delta t} & \text{if} \quad \frac{\angle F_r(\mathbf{q}_i)}{\Delta t} < w_{max} \\ \pm w_{max} & \text{if} \quad \frac{\angle F_r(\mathbf{q}_i)}{\Delta t} \geq w_{max} \end{cases} \tag{9}$$

where $\angle F_r(\mathbf{q}_i)$ is the orientation of the force, and w_{max} is the max angular velocity. The sign of w_i is chosen to rotate the robotic sensor to the orientation of force in the shorter angle, and the velocity command is,

$$v_i^c = v_i + k\|F_r(\mathbf{q_i})\| \tag{10}$$

where v_i is current velocity, and k is a constant. In the practical case, v_i is bounded by the maximum velocity v_{max}. Therefore, v_i^c is pruned to v_{max} if v_i^c is greater than v_{max}. The v_i^c and w_i^c are sent to low-level controller embedded in robot model of Gazebo. When $N_r(\mathbf{q}) = \emptyset$, the modified RRT method presented in Section IV-B is used to compute the robotic sensor control.

When the robotic sensor is in the exploitation state, the set of its neighbors is defined as $N_t(\mathbf{q}_i) = \{\rho(\mathbf{q}_i,\mathbf{q}_j) < \rho_0,\ \forall j \in I_t, j \neq i\}$, and its artificial potential field is constructed as follows,

$$U_t(\mathbf{q}_i) = \sum_{j\in N_t(\mathbf{q}_i)} U_j(\mathbf{q}_i) + \sum_{j\in I_B} U_j(\mathbf{q}_i) \tag{11}$$

When $N_t \neq \emptyset$, the control for the ith robotic sensor is computed similarly to (9) and (10). If $N_t(\mathbf{q}) = \emptyset$, the modified RRT method presented in Section IV-B is used to compute the robotic sensor control. The process ends when a predefined number of targets are measured by high accurate sensors, or the running time reaches a defined threshold.

B. RRT Online Sensor Path Planning

A novel RRT approach is presented for online geometric sensor path planning in partially-observed environments. In this approach, the milestone sampling and tree expansion are based on the expected information value of targets that have been detected by an airborne or ground robotic sensor, the distance between the robotic sensor platform and the obstacles, and the distance between the sensor's FOV and the targets.

1) Milestone sampling: The approximate FOV is based on the ability of the sensor to emit and receive the reflex signal from different directions, such that the distance between the robotic sensor and obstacles within its FOV can be computed. Let $L(\mathbf{q}) = (l_1, l_2, \ldots, l_n) \in \Re^n$ represent the distance between the obstacles and the robotic sensor. When no reflex is found for an orientation, the corresponding distance is set as the sensor range. Then assume that the distribution of sample orientation is a mixture normal distribution with n components. Each normal distribution corresponds to an orientation of the reflex signal. For the ith robotic sensor, we have,

$$\theta_i^s \sim \sum_{j=1}^{n} m_i^j N(\mu_i^j, \sigma_{1i}^2) \tag{12}$$

where m_i^j is the weight for the jth normal distribution, μ_i^j is the mean and is set to the direction of jth reflex, and σ_{1i} is the standard deviation and is set to,

$$\sigma_{1i} = \frac{a v_{max}}{v_i + b v_{max}} \tag{13}$$

where a and b are constant parameters, v_{max} is the max allowed velocity for the robotic sensor, and v_i is its current linear velocity. By setting σ_{1i} in this way, larger velocity decreases the spread of the sample.

The weight m_i^j is computed as,

$$m_i^j = \frac{l_i^j}{\sum_{j=1}^{n} l_i^j} \tag{14}$$

When a safety distance l_0 is utilized to avoid collision, l_i^j is set to zero when $l_i^j \leq l_0$. It can be seen that the direction with lower l_i^j has a lower weight m_i^j. This results in a lower probability of sampling configurations along the corresponding orientation, and then can help the robotic sensor to move to collision free region. By setting l_0, it is possible that

the robotic sensor can take a measurement of target near to obstacles. With this definition, our method is able to adjust the geometry of $\mathcal{C}_{free}$ automatically online without adjust the parameters or set multiple sample strategies for different situations.

The sampled θ_i^s is utilized in (2) to generate the samples for the current tree. In our case, we set σ_{2i} in (3) as,

$$\sigma_{2i} = \frac{cv_i}{v_i + dv_{max}} \tag{15}$$

Here, c and d are constant parameters. When v_i is large, σ_{2i} prefers to be big which may give a bigger sample distance. In the simulation with Gazebo, the distance and direction of obstacles around the agent is available from the low accurate sensor. Then this information can be used to biased θ_i^s to a region that navigates the robotic agent to avoid the obstacle region.

After a number of milestones are sampled for the ith sensor, they are ordered based on their important value based on the robotic sensor's state. When the sensor is in exploration state, the value of a sample is defined as,

$$R(\mathbf{q}) = \rho_i(\mathbf{q}) \tag{16}$$

where $\rho_i(\mathbf{q})$ is the distance between $\mathbf{q}$ and the agent. The sensor prefers to choose a milestone that is far away to its current configuration which is consistent to the purpose of exploration.

For a sensor in the exploitation state,

$$R(\mathbf{q}) = k_2 e^{-\frac{1}{2f\rho_i(\mathbf{q})^2}} + k_1 \sum_{j \in N_i} e^{-\frac{\rho_j^t(\mathbf{q})^2}{2eV(j)^2}} \tag{17}$$

where k_1 and k_2 are two constant representing the weight, N_i is the index of targets that is assigned to the ith robotic sensor, $\rho_j^t(\mathbf{q})$ is the distance between $\mathbf{q}$ and $\mathcal{CT}_j$, and $V(j)$ is the expected information benefit of the jth target. It can be seen that $R(\mathbf{q})$ is a increasing function of $V(j)$ and $\rho_i(\mathbf{q})$, and is a decreasing function of $\rho_j^t(\mathbf{q})$. So the sampler prefers to generate a sample with big distance to its current configuration and small distance to the targets assigned to it. By differentiating $R(\mathbf{q})$ to ρ_j^t we have,

$$\frac{\partial^2 R}{\partial \rho_j^{t2}} = \frac{k_1 e^{-\frac{\rho_j^t(\mathbf{q})^2}{2eV(j)^2}}}{eV(j)^2}\left(\frac{\rho_j^t(\mathbf{q})^2}{eV(j)^2} - 1\right) \tag{18}$$

Then,

$$\frac{\partial^2 R}{\partial \rho_j^{t2}} = 0 \Rightarrow \rho_j^t = \sqrt{e}V(j) \tag{19}$$

So the target's expected information benefit determines the influence distance to effect R. Similarly by differentiating $R(\mathbf{q})$ to ρ_i we can see the influence distance between the sample and the robotic sensor configuration is $\rho_i = \sqrt{f}$. The ordered samples then are used to expand the tree as described in the next subsection.

2) Tree expansion: During online sensor path planning, no global RRT exists for each robotic sensor. A local RRT is constructed and updated for each robotic sensor during its movement. Since the passed path has no use for the robotic sensor, all nodes connected to a node that has been passed by the robotic sensor are deleted and the root for the tree constructed at next time step is set as the node the robotic sensor moves towards. The tree is updated when the distance from the robotic sensor to the new root is smaller than a threshold ϵ.

The update process contains three steps. At the first step a number of configurations are sampled which are ordered in expectation descended by considering target and obstacle locations, and expected information benefit of targets as discussed in above subsection. Then the sample configuration feasibility is checked by computing the expected path to the selected sample, i.e., to check whether the expected path lies in $\mathcal{C}_{free}$. The expected path is computed as follows. Let the vector $F^s(\mathbf{q}_i)$ be the vector pointing from the agent i's current configuration to the selected sample. Define,

$$\gamma(\mathbf{q}_i) = \min(\angle F^s(\mathbf{q}_i), \theta_i) \tag{20}$$

where $\min(\alpha, \beta)$ gives the shortest angle to rotate a agent heading angle from β to α. Then the angular velocity control is defined as,

$$w_i^c = \begin{cases} \pm a\gamma(\mathbf{q}_i) & \text{if} \quad a\gamma(\mathbf{q}_i) < w_{max} \\ \pm w_{max} & \text{if} \quad a\gamma(\mathbf{q}_i) \geq w_{max} \end{cases} \tag{21}$$

where a is a constant parameter, and the sign of w_i^c is chosen to rotate the robotic sensor to the orientation of force with the shorter angle. The angular velocity control is defined as,

$$v_i^c = \frac{e}{f + \frac{h\gamma(\mathbf{q}_i)}{l + m\rho(\mathbf{q}_i)}} \tag{22}$$

where e, f, h, l, and m are positive constants, and $\rho(\mathbf{q}_i)$ is the distance from the agent i to the selected sampled configuration. From (22) we can see that v_i^c has the upper bound as $\frac{e}{f}$ and is a decreasing function of $\gamma(\mathbf{q}_i)$, i.e., the bigger the angular velocity control, the lower the linear velocity command. Furthermore, v_i^c is an increasing function of $\rho(\mathbf{q}_i)$, which means that the further the distance between the selected sample and the robotic sensor, the higher the linear velocity command. The reason to set the robotic sensor control in this way is to prevent the robotic sensor turning over along the path. A centrifugal acceleration max value, g, is further utilized to guarantee the movement of the robotic sensor, i.e., when the computed v_i^c (or w_i^c) from (22) is bigger than $\frac{g}{w_i}$ (or $\frac{g}{v_i}$), it is pruned to $\frac{g}{w_i}$ (or $\frac{g}{v_i}$), where w_i and v_i is the current angular and linear velocity. Then v_i^c and w_i^c are used as the control commands for (4) to compute the expected path.

At last, the feasible configuration with the highest value as computed in (16) or (17) is set as the child of the current configuration.

After the tree for next time step is updated, the control are computed in (21) and (22) and then utilized to control the robotic sensor movement in Gazebo.

3) Decidability with Differential Constraints: When expanding a RRT tree, the most time consuming part is to check

whether there exists a free path to a milestone. This leads to another problem, the decidability of motion planning with differential constraints (MPD) [27] for the Pioneer car model, i.e. to decide in finite time whether the sampled milestone can be reached. In this section, the above MDP problem is proved to be decidable under proper assumptions. We assume that the control of the Pioneer car to milestone can be divided into k control steps in sequence. At each step, we assume that the velocity command and turning rate command of the Pioneer car can only be constant v^c and w^c under control $\mathbf{u}$, where $\mathbf{u} \in \mathcal{U}$, $v^c \in \mathcal{U}_v$, and $w^c \in \mathcal{U}_w$. $\mathcal{U}_v$ is a discrete set defined as,

$$\mathcal{U}_v = \{v_1,\ v_2,\ \cdots, v_r\} \tag{23}$$

where v_i is a value in velocity interval $[0,\ v_{max}]$ and r is the cardinality of set $\mathcal{U}_v$. $\mathcal{U}_w$ is a discrete set defined as,

$$\mathcal{U}_w = \{w_1,\ w_2,\ \cdots,\ w_h\} \tag{24}$$

where w_i is a value in turning rate interval $[-w_{max},\ w_{max}]$ and h is the cardinality of set $\mathcal{U}_w$. $\mathcal{U}$ is a discrete set defined as,

$$\mathcal{U} = \{\mathbf{u}_1,\ \mathbf{u}_2,\ \cdots,\ \mathbf{u}_d\} \tag{25}$$

where $\mathbf{u}_i$ is a single step control, which specifies the linear velocity v^c and angular velocity w^c, and d is the cardinality of the control space, which is $r \times h$. Also, we assume that the v and w of the Pioneer car can converge to v^c and w^c quickly enough so that v and w are constant under the single step control $\mathbf{u}$. With the above assumptions, the Pioneer car is approximated to a dubin's car [28]. The k-steps control , $\overline{\mathbf{u}}$, is defined as a "concatenation" of k elements from $\mathcal{U}$. Let $\mathcal{U}^k$ denote the k-steps control space and defined as,

$$\mathcal{U}^k = \{\overline{\mathbf{u}} \mid \overline{\mathbf{u}} = \{\mathbf{u}^1,\ \mathbf{u}^2, \cdots, \mathbf{u}^k\},\ \mathbf{u}^i \in \mathcal{U},\ i = 1,\ 2, \cdots k\} \tag{26}$$

The cardinality of $\mathcal{U}^k$ is $r \times h \times k$. Then the obstacles with polygonal shapes can be represented by semi-algebraic sets $\mathcal{C}_{obs}$ which are defined as the following set [27],

$$\{\mathbf{q} | \vee_{i=1,2,\cdots,n_o} (\wedge_{m=1,2,\cdots,n_e^i} H_{i,m}(\mathbf{q})) \leq 0\}, \tag{27}$$

in which $\vee$ and $\wedge$ are boolean operators "or" and "and", $H_{i,m}$ is polynomial function of $\mathbf{q}$, n_o is the number of obstacles and n_e^i is the edge number of ith obstacle. Let $\mathbf{q}_m$ denote a milestone configuration and let $\mathcal{TR}(\overline{\mathbf{u}})$ denote the trajectory by control $\overline{\mathbf{u}} \in \mathcal{U}^k$. Since the trajectory function $\mathcal{TR}(\mathbf{u})$ by one step control $\mathbf{u}$ is a closed form polynomial function and $\mathcal{TR}(\overline{\mathbf{u}})$ is the"concatenation" of $\mathcal{TR}(\mathbf{u})$, $\mathcal{TR}(\overline{\mathbf{u}})$ is also closed form polynomial function [27]. We define the set,

$$\mathcal{U}_{free}^k = \{\overline{\mathbf{u}} \in \mathcal{U}^k \mid \mathcal{TR}(\overline{\mathbf{u}}) \cap \mathcal{C}_{obs} = \phi\} \tag{28}$$

as the control subset free of collision in $\mathcal{U}^k$ and the set,

$$\mathcal{U}_m^k = \{\overline{\mathbf{u}} \in \mathcal{U}^k \mid \mathcal{TR}(\overline{\mathbf{u}}) \cap \mathbf{q}_m \neq \phi\} \tag{29}$$

as all controls that lead the sensor to reach a milestone. The control set which can lead a Pioneer car to reach a milestone is defined as $\mathcal{U}_{goal}^k = \mathcal{U}_{free}^k \cap \mathcal{U}_m^k$. Based on above analysis, the trajectory $\mathcal{TR}(\overline{\mathbf{u}})$ is a closed form polynomial function. Furthermore, since $\mathcal{C}_{obs}$ is semi-algebraic, the two sets $\mathcal{U}_{free}^k$, and $\mathcal{U}_m^k$ are also semi-algebraic according to Tarski's theorem. Therefore, the MDP problem for Pioneer car to a milestone under above assumptions is decidable [27].

V. Software Implementation

In this paper, Matlab and Gazebo are interfaced to model and control the robotic sensor network in the three-dimension space. Simulation system Gazebo (originally developed at USC robotics research lab) runs as a server, which maintains all the robots, sensors, targets, map models, and the physical relationships between all objects. The client program used to control robots is individually connected to the server, and it can get information of robots and sensors through the interface of Gazebo. While Gazebo and parts of the client program are written in C++, the control model is written in Matlab. Therefore, an interface between C++ and Matlab is needed. We use the functions in the dynamic link library called *engine.so*, such as *engEvalString*, *engGetPr* and *engPutVariable* to communicate Gazebo coding environment with Matlab.

As shown in Fig. 3, the supercomputer is coded in Matlab

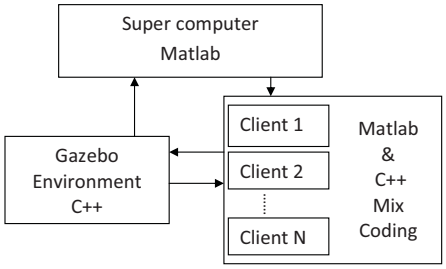

Fig. 3. The flow chart between modules of the coding system.

which obtains robot configuration and velocity information from Gazebo environment, and allocates tasks to clients separately. The client program is coded in both C++ and Matlab, and has an interface with Gazebo. With this interface, a client program can obtain the simulation data and then control a unique robot by sending linear and angular velocity commands to a lower-level feedback controller of the robot. Furthermore, the client program has access to all the information of this robot, such as its configuration and velocity. Two types of sensors are used in this paper. The first sensor is with limited sensor FOV but can detect targets with high accuracy; the other can emit two frequency waves. One frequency can detect targets with low accuracy but wide FOV, and the other frequency can be reflected by the obstacles to detect the distance to obstacles. Measurements of target detection waves are modeled by Bayesian network [24], which is coded in Matlab. The server written in C++ can interface with Matlab, input all parameters that the sensor model needs to simulate sensor measurement, and obtain simulated measurement results. The robot used in the simulation is based on PIONEER2DX and is equipped with the sensor described above. The robot utilizes a differential

controller which calculates the command velocities of both wheels according the linear velocity and angular velocity commands.

VI. Simulations and Results

In this section, a number of simulations are implemented to test the proposed method. It is assumed that 12 targets and 11 obstacles are deployed in a square workspace with side length of 50 meters. The high accurate sensor FOV $\mathcal{S}$ is assumed to be a cone, and the low accurate sensor FOV $\mathcal{G}$ is assumed as a set of half circle sharing the same diameter. Each target and obstacle are assumed to be a polyhedron whose cross section $\mathcal{H}_z$ by the plane $Z = z$, has the same geometry and location for all z satisfying $\mathcal{H}_z \neq \emptyset$. With this assumption, the sensor FOV could be reduced to a two dimensional polygon that varies based on the target geometry and location. An example of the reduced sensor FOV for a target is shown in Fig. 4. In this figure, the sensor FOV apex position in $\mathcal{F}_\mathcal{A}$ is $(0, 0, 5)$, and the target is assumed to be float in the air with minimal height is 6.

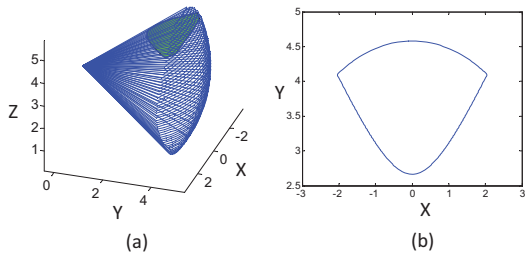

Fig. 4. The Reduced FOV of a sensor with taper shape: (a) sensor FOV in three dimensional space; (b) reduced sensor FOV.

The method similar to the one in [5] can be used to compute the regions of robotic sensor configuration in $\mathcal{C}$ that has intersection with obstacles or targets called C-obstacles or C-targets. An example of the workspace is shown in Fig. 5.

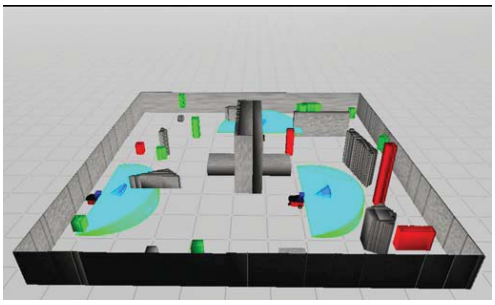

Fig. 5. A simulation system in Gazebo: gray objects represent obstacles and boundaries; the red objects represent targets which are grounded; the green objects represent targets which are in the air; the robot has two sensors, one is high accuracy with small range, another one is low accuracy with large range.

In the simulation, we assume that prior information on a portion of obstacles and targets is known, and the purpose of the robotic sensor group is to measure and classify six targets. An example of the predicted path for the robotic sensor moving from the current configuration to the sample is shown in Fig. 6.

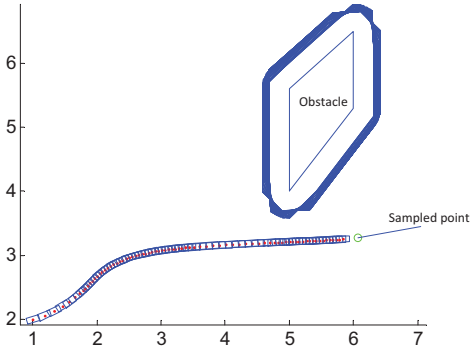

Fig. 6. A sensor path from the current configuration to the sampled configuration.

The performance of each simulation is represented by its efficiency η, and is computed as,

$$\eta = \frac{W_1 - W_0}{D} \tag{30}$$

where W_1 is the number of targets that are correctly classified at the end, W_0 is the number of targets that are correctly classified as the beginning, and D is the total distance traveled by the robotic sensor group. η gives us the reward of the robotic sensor group when they travel a unit distance [7]. Each method runs ten times and the average results are shown in Table I. The third row in Table I shows results of a random search method, in which each robotic sensor movement is controlled in a similar way but the expected information value was not considered.

TABLE I
The results of the efficiency of the robotic sensor group by the proposed method with and without utilizing prior information

Use Information	Known $\mathcal{T}$	Known $\mathcal{B}$	η
Yes	All	All	0.043
Yes	None	All	0.038
No	All	All	0.020

Among these simulations, two simulations utilize the prior information to navigate the robotic sensor. In all cases, we assume the geometries of all obstacles are known *a priori*. From the results, we can see that in all three cases, the proposed hybrid system can successfully utilize the high accurate sensor to improve the classification correctness of the targets in a field. Moreover, by utilizing the prior information and information gathering from a low accurate sensor along the process, the efficiency of the robotic sensor group is significantly improved comparing to the one without utilizing prior information. A path of our method is shown in Fig. 7.

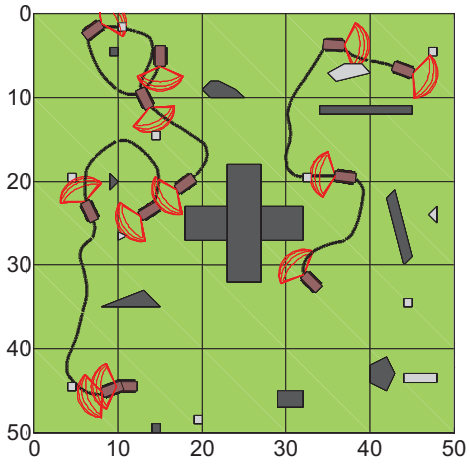

Fig. 7. An example of the sensor group path. black area: obstacle, grey area: target, dark yellow rectangle: sensor platform, red line: high accurate sensor FOV (The geometry of low accurate sensor FOV is eliminated for display).

VII. CONCLUSIONS

A hybrid system is proposed for a network of robotic sensors in searching and measuring targets in a partial observed environment containing multiple obstacles and multiple targets. A modified rapidly-exploring random tree (RRTs) method is designed for planning the path of the robotic sensor network online. The proposed method is tested with a number of simulations implemented with the software environment Gazebo, and the simulation results show that the robotic sensor network can measure the targets based on the trade off on distance and the target's information value.

The future work will lie in the following three parts. First, the properties of the proposed hybrid system, such as convergent, completeness, and computation complexity will be studied. Second, heterogeneous sensors, for instance unmanned aerial vehicles, will be included to perform as a source of obtaining prior information on the targets with cursory measurements. Finally, besides stationary targets, moving targets will be considered in the workspace, and their movement is assumed to be partially known and are inferred along the process.

VIII. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation, under grant NSF ECS CAREER 0448906.

REFERENCES

[1] S. Y. Chen and Y. F. Li, "Automatic sensor placement for model-based robot vision," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. 34, no. 1, pp. 393–408, 2004.

[2] S. Chen and Y. Li, "Vision sensor planning for 3cd model acquisition," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. 35, no. 5, pp. 894–904, 2005.

[3] H. Choset, "Coverage for robotics: A survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1-4, pp. 113–126, 2001.

[4] K. Baumgartner and S. Ferrari, "A geometric transversal approach to anayzing track coverage in sensor networks," *IEEE Transactions on Computers*, vol. 57, no. 8, 2008.

[5] C. Cai and S. Ferrari, "Information-driven sensor path planning by approximate cell decomposition," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. 39, no. 3, pp. 607–625, 2009.

[6] 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.

[7] G. Zhang, S. Ferrari, and M. Qian, "Information roadmap method for robotic sensor path planning," *Journal of Intelligent and Robotic Systems*, vol. 56, no. 1-2, pp. 69–98, 2009.

[8] G.Zhang and S.Ferrari, "An adaptive artificial potential function approach for geometric sensing," in *Proc. of IEEE International Conference on Decision and Control*, Shanghai, China, 2009, pp. 7903–7910.

[9] R. Siegel, "Land mine detection," *IEEE Instrumentation and Measurement Magazine*, vol. 5, no. 4, pp. 22–28, 2002.

[10] C. Hofner and G. Schmidt, "Path planning and guidance techniques for an autonomous mobile cleaning robot," *Robotics and Autonomous Systems*, vol. 14, pp. 199–212, 1995.

[11] S. Ferrari, C. Cai, R. Fierro, and B. Perteet, "A multi-objective optimization approach to detecting and tracking dynamic targets in pursuit-evasion games," in *Proc. of the 2007 American Control Conference*, New York, NY, 2007, pp. 5316–5321.

[12] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[13] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet," *Proc. $10^t h$ International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X)*, 2002.

[14] E. U. Acar, "Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods," *International Journal of Robotic Research*, vol. 22, 2003.

[15] 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. 39, no. 3, pp. 607–625, 2009.

[16] L. E. Kavraki, P. Svetska, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration space," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[17] N. M. Amato and Y. Wu, "A randomized roadmap method for path and manipulation planning," in *In IEEE Int. Conf. Robot. & Autom*, 1996, pp. 113–120.

[18] J. J. K. Jr. and S. M. Lavalle, "Rrt-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2000, pp. 995–1001.

[19] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[20] "The player project," Website, http://playerstage.sourceforge.net.

[21] K. Kastella, "Discrimination gain to optimize detection and classification," *IEEE Transactions on Systems, Man, and Cybernetics - Part A*, vol. 27, no. 1, pp. 112–116, 1997.

[22] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal Processing Magazine*, vol. 19, pp. 61–72, 2002.

[23] C. Cai, S. Ferrari, and Q. Ming, "Bayesian network modeling of acoustic sensor measurements," in *Proc. IEEE Sensors*, Atlanta, GA, 2007, pp. 345–348.

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

[25] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep., 1998.

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

[27] P. Cheng, G. Pappas, and V. Kumar, "Decidability of motion planning with differential contraints," in *Proceedings of the 46nd IEEE International Conference on Robotics and Automation*, Roma Italy, 2007, pp. 1826–1831.

[28] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.