Maintaining Robust Connectivity in Heterogeneous Robotic Networks

P. Cruz[§], R. Fierro[§], W. Lu[†], and S. Ferrari[†]

[§]Multi-Agent, Robotics, Hybrid and Embedded Systems (MARHES) Laboratory,
Department of Electrical and Computer Engineering,
University of New Mexico, Albuquerque, NM, USA;
[†]Laboratory of Intelligent Systems and Control (LISC),
Department of Mechanical Engineering and Materials Science,
Duke University, Durham, NC, USA

ABSTRACT

In this paper, we are interested in exploiting the heterogeneity of a robotic network made of ground and aerial agents to sense multiple targets in a cluttered environment. Maintaining wireless communication on this type of networks is fundamentally important specially for cooperative purposes. The proposed heterogeneous network consists of ground sensors, *e.g.*, OctoRoACHes, and aerial routers, *e.g.*, quadrotors. Adaptive potential field methods are used to coordinate the ground mobile sensors. Moreover, a reward function for the aerial mobile wireless routers is formulated to guarantee communication coverage among the ground sensors and a fixed base station. A sub-optimal controller is proposed based on an approximate control policy iteration technique. Simulation results of a case study are presented to illustrate the proposed methodology.

Keywords: Robust Connectivity, Adaptive Potential Functions, Approximate Policy Iteration, Heterogeneous Systems

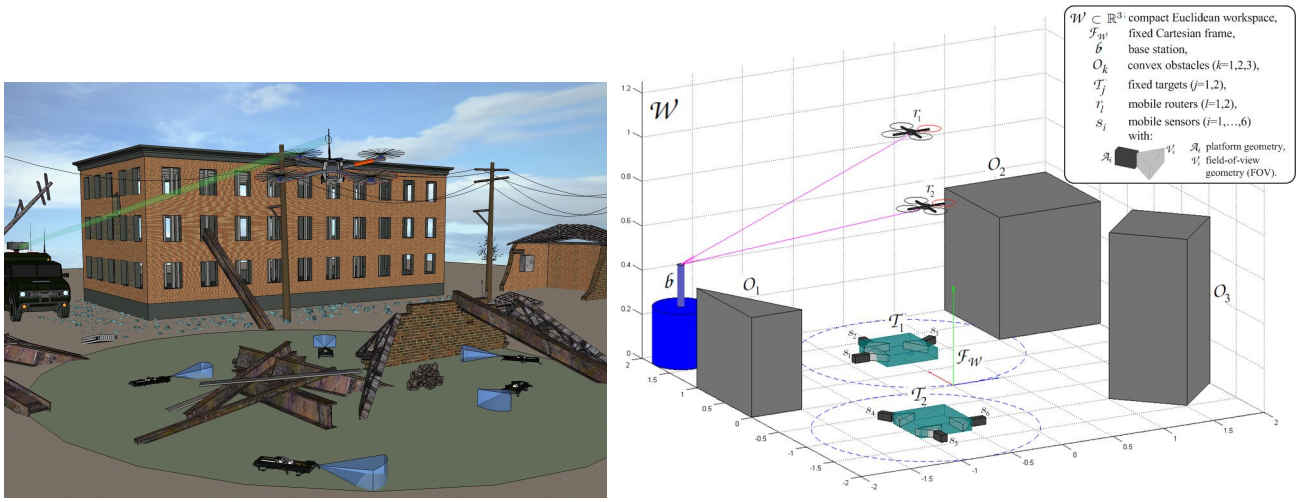
1. INTRODUCTION

The scenario envisioned in this paper is intended for target sensing in hazardous environments like in a disaster area with collapsed structures, see Figure 1(a). In these situations, the use of heterogeneous systems made up of aerial and ground robotic vehicles would maximize the probability to efficiently and successfully accomplish the mission. For instance, aerial vehicles have the capability to cover an area faster but cannot have a detailed view of caves or buildings. On the other hand, multi-legged crawling robotic platforms can only explore a limited area, but do so with much more accuracy. In addition, a reliable wireless connectivity is an important factor to be considered when dealing with multi-agent systems. Due to several limitations in the communication channel, especially when the transmission is through the air medium, complications such as shadow effects and secondary reflections arise. These phenomena create a variety of constrains on the possible relative positions of the heterogeneous agents. Thus, we are interested in developing strategies to enhance connectivity of the network of robots and a fixed base station while a given number of targets are sensed. To be more specific, we describe a target sensing algorithm for a ground mobile sensor network by relaxing the assumption of network connectivity introducing a specialized aerial router agents which are better equipped to communicate over longer distances. In order to solve this problem, the mobile sensors are coordinated based on an adaptive potential method; while, *Online Least-Squares Policy Iteration* (online LSPI)[1,2], an *Approximate Dynamic Programming* (ADP) policy iteration technique, is used to modify the position of the mobile routers in order to maintain the communication capabilities between the sensor network and the base station.

The objective of maintaining connectivity together with additional requirements like collision avoidance or formation control in the motion planning of a multi-robot system has been extensively studied in the literature.

Further author information: (Send correspondence to P. Cruz)
[§] P. Cruz and R. Fierro: {pcruzec, rfierro}@ece.unm.edu
[†] W. Lu and S. Ferrari: {wenjie.lu, sferrari}@duke.edu

Unmanned Systems Technology XV, edited by Robert E. Karlsen, Douglas W. Gage,
Charles M. Shoemaker, Grant R. Gerhart, Proc. of SPIE Vol. 8741, 87410N
 · doi: 10.1117/12.2016236



(a) (b)

Figure 1. (a) A quadrotor as communication router for a team of crawlers that are exploring and sensing a disaster area; and (b) the 3D simulation environment used in this work with a legend describing its components. The magenta vector from each one of the mobile routers to the base station indicates there exists a point-to-point link between these two elements at any time.

For instance in[3], decentralized controllers based on navigation functions for a group of robots are developed to satisfy individual sensing goals while some neighborhood connectivity relationships are maintained. In[4], communication range and line-of-sight are used as motion constraints for a swarm of point robots which goes from an initial to a final configuration in a cluttered environment. Meanwhile,[5] introduces a multi-robot exploration algorithm that uses a utility function built taking into account the constraints of wireless networking. Monitoring the communication link quality or the construction of a signal strength map are strategies described in[6] for a good link quality maintenance in the deployment of a mobile robot network. Stochastic communication models are used in[7] to develop an algorithm to maintain end-to-end connectivity in a team of autonomous robots.

Most of the literature on multi-agent robotic systems that considers some type of communication constraint assumes homogeneity among the members of the system. However, recent works[8–11] have started exploiting the advantage of a heterogeneous multi-robot network to enhance the communication capabilities of the whole system. In,[8] a team of Unmanned Ground Vehicles (UGVs) performs a collaborative task while a team of Unmanned Aerial Vehicles (UAVs) is positioned in a configuration such that they optimize the communication link quality to support the team of UGVs, but the authors assume that the UGVs are static to guarantee the connectivity of the UAV-UGV network. By using a heterogeneous robotic system, a search/pursuit scenario is implemented in[9] where a control algorithm guarantees a certain level of Signal-to-Interference plus Noise Ratio (SINR) among the members of the system. Even though the field-of-view of the sensor members in the network is considered, the geometry of the agents is neglected. The authors in[10] introduce a mobile communication relay to a network of sensors and derive connectivity constraints among the network members. Afterwards, these constraints are used to maximize the feasible motion sets of the sensing agents. In this work as assumption, the network moves in a free-obstacle environment and its agents are considered as point robots. Communication maintenance for a group of heterogeneous robots is enforced in[11] by a passivity-based decentralized strategy. This approach allows creation/deletion of communication links at any time as long as global connectivity is preserved, but the strategy is not tested in the case that the network has a main goal like sensing on top of maintaining connectivity.

Our goal is to deploy a heterogeneous system made of mobile sensors and a mobile router in an obstacle-populated area to get measurements of a number of fixed static targets. Moreover, we consider the platform and field-of-view geometry[12] for the case of the sensors; while, we assume point robots only for the case of the routers. In addition, we present the 3-D environment implemented in MATLAB for simulation purposes.

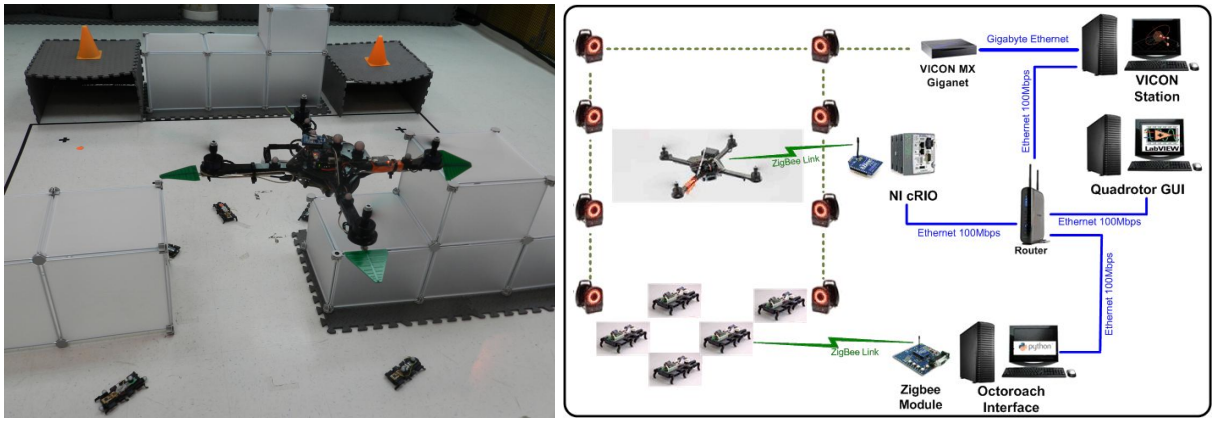
2. BACKGROUND

In this section, we present a concise explanation about the concept of heterogeneity in robotic networks. Also, we provide a brief overview about potential field methods and approximate dynamic programming that are behind the implementation of the controller that will be discussed in Section 4.

2.1 Heterogeneous Robotic Networks

Heterogeneity is used in general to describe a network that consists of agents with variations on their hardware structure and/or on their mission objective[10]. For example, hardware heterogeneity includes different sensor footprints,[13] different communication ranges[10] and of course different agent dynamics[9], among many others. On the other hand in objective-based heterogeneity, the overall mission goal is divided into multiple sub-tasks that are assigned to each one of the agents in the network. A clear example of heterogeneity based on the mission objective is the dynamic sub-task coordination of an heterogeneous robotic team in the RoboCup soccer competitions[14]. The difficulty in deriving suitable controllers for heterogeneous robotic networks lies in handle the hardware variations to combine behaviors in a way that the overall objective is achieved maintaining properties such as convergence and stability.

The Multi-Agent Robotics Hybrid and Embedded Systems Laboratory, MARHES Lab, at the University of New Mexico is a clear example of an heterogeneous robotic network[15]. Its robotic test-bed consists of holonomic and non-holonomic ground vehicles, rotorcraft unmaned aerial vehicles, small crawling robots and other commercially available platforms. Figure 2(a) shows an aerial vehicle hovering over a group of legged-crawling robots in a cluttered environment. In addition, the laboratory is equipped with a high-precision motion capture system, an up-to-date sensor suite, high-performance embedded controllers and wireless communication devices for networking. The block diagram in Figure 2(b) presents an example of how the test-bed is setup from the communications perspective. The MARHES Lab falls under both hardware-based and objective-based heterogeneity since its robotic platforms present different agent dynamics and for example the ground robots can be considered as mobile sensors while the aerial agents can act as communication routers dividing efforts to accomplish a common main goal.



(a) (b)

Figure 2. The MARHES Lab, an heterogeneous robotic network: (a) a quadrotor hovering over a group of small crawling robots, and (b) a block diagram showing the communication links for experimental purposes.

2.2 Potential Field Methods

The potential field method is a robot motion planning approach that controls a robot movement based on the gradient field of a potential function[12]. Potential field was originally developed as an on-line collision avoidance approach, applicable when a robot does not have a prior model of the obstacles, but senses them during motion execution[16]. The main idea besides most proposed potential functions[12,16,17] is that a robot should be attracted

toward its target configuration, while being repulsed by possible obstacles. Therefore, the obstacle and target configuration are considered as sources to construct a potential function U. In general, U consists of two components: an attractive potential U_{att} generated for example by the target configuration and a repulsive potential U_{rep} generated for example by the obstacles. Thus, the total potential is given by

$$U(\mathbf{x}) = U_{att}(\mathbf{x}) + U_{rep}(\mathbf{x}),$$

where $\mathbf{x} = [x_1 \ x_2 \ ... \ x_n]^T \in \mathbb{R}^n$ is the configuration state of the robot. The force applied on the robor is proportional to the negative gradient of U

$$\nabla U(\mathbf{x}) = \left[\frac{\partial U((x))}{\partial x_1} \ \frac{\partial U((x))}{\partial x_2} \ ... \ \frac{\partial U((x))}{\partial x_n} \right]^T,$$

and it is used to design a controller for the robot movement. As in,[12,16] an attractive potential can be represented as

$$U_{att}(\mathbf{x}) = \frac{1}{2}\eta_{att}\varrho_t^2(\mathbf{x}), \tag{1}$$

where η_{att} is a scaling factor and $\varrho_t(\mathbf{x})$ is the Euclidean distance between the robot and the target configuration. Meanwhile, a repulsive potential can be given by

$$U_{rep}(\mathbf{x}) = \begin{cases} \frac{1}{2}\eta_{rep}\left(\frac{1}{\varrho_o(\mathbf{x})} - \frac{1}{d_0}\right)^2 & \text{if } \varrho_o(\mathbf{x}) \leq d_0, \\ 0 & \text{if } \varrho_o(\mathbf{x}) > d_0, \end{cases} \tag{2}$$

where η_{rep} is a scaling factor, $\varrho_o(\mathbf{x})$ is the Euclidean distance between the robot and the nearest obstacle, and d_o is the distance of influence of the obstacles. In particular, an on-line potential field method essentially acts as a descent optimization procedure, so it may get stuck at a local minimum other than the goal configuration. However, the combination of potential field methods with graph searching techniques has demonstrated to be a valid approach such that a robot escapes the local minimum[12,18,19].

2.3 Approximate Dynamic Programming

In recent years, *Reinforcement Learning*, RL, has grown as an effective approach for control applications and indeed many new formulations under the RL umbrella have appeared[20]. RL refers to an actor, *e.g.*, a robotic agent, which interacts with its environment and modifies its actions, *e.g.*, its control policy, based on stimuli received in response to its actions, *e.g.*, a scalar reward value. *Approximate Dynamic Programming*, ADP, is a family of techniques within RL for the feedback control of human engineered systems[21,22]. Indeed, ADP is an effective framework to circumvent the curse of dimensionality in *Dynamic Programming*, DP, preventing the direct adoption of DP in many real-world control problems[1,22–25]. Now, consider the discrete-time nonlinear deterministic system

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k),$$

where k is the discrete time, $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$ represents the state vector of the system, $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$ denotes the control action and $f : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ is the system function. The control policy for this system is defined as a function $h : \mathcal{X} \to \mathcal{U}$ such that

$$\mathbf{u}_k = h(\mathbf{x}_k),$$

and a reward function $\rho : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ evaluates the immediate effect of the action $\mathbf{u}_k$. Indeed, it produces a scalar reward signal given by

$$\Upsilon_k = \rho(\mathbf{x}_k, h(\mathbf{x}_k)).$$

f and ρ together with $\mathcal{X}$ and $\mathcal{U}$ constitute a so-called *Markov Decision Process*, MDP[1,23]. Indeed, given f, ρ, the current state $\mathbf{x}_k$ and the current action $\mathbf{u}_k$ are sufficient to find the next state $\mathbf{x}_{k+1}$ and the reward Υ_k. This is the well known Markov property[1,24].

The return Q-function $Q^{(h)} : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ represents the reward obtained by the controller in the long run and it is given by

$$Q^{(h)}(\mathbf{x}_k, \mathbf{u}_k) = \sum_{i=k}^{\infty} \gamma^{i-k} \rho(\mathbf{x}_i, \mathbf{u}_i), \tag{3}$$

where $\gamma \in [0, 1)$ is a discount factor. In RL and DP the goal is to find an optimal policy that maximizes the return. Rewriting (3) as

$$Q^{(h)}(\mathbf{x}_k, \mathbf{u}_k) = \rho(\mathbf{x}_k, \mathbf{u}_k) + \gamma \sum_{i=k+1}^{\infty} \gamma^{i-(k+1)} \rho(\mathbf{x}_i, \mathbf{u}_i),$$

then its equivalent difference equation is

$$Q^{(h)}(\mathbf{x}_k, \mathbf{u}_k) = \rho(\mathbf{x}_k, \mathbf{u}_k) + \gamma Q^{(h)}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}), Q(\mathbf{x}_0, \mathbf{u}_0) = 0.$$

This is a nonlinear Lyapunov equation known as Bellman equation and its optimal value can be written as[21]

$$Q^*(\mathbf{x}_k, \mathbf{u}_k) = \max_h \left\{ \rho(\mathbf{x}_k, \mathbf{u}_k) + \gamma Q^{(h)}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \right\}.$$

By Bellman's optimality principle, one gets that

$$Q^*(\mathbf{x}_k, \mathbf{u}_k) = \max_h \left\{ \rho(\mathbf{x}_k, \mathbf{u}_k) + \gamma Q^*(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \right\}.$$

This is the Bellman optimality equation also known as the discrte-time Hamilton-Jacobi-Bellman (HJB) equation. Then, the optimal control policy at time k is equal to

$$h^*(\mathbf{x}_k) = \arg\max_h \left\{ \rho(\mathbf{x}_k, h(\mathbf{x}_k)) + \gamma Q^*(\mathbf{x}_{k+1}, h(\mathbf{x}_{k+1})) \right\}. \tag{4}$$

Notice that the optimal policy at $k + 1$ must be known to determine the optimal policy at time k. Thus, Bellman's principle yields by nature to offline planning methods. Also, the classical DP and RL algorithms require exact representation of the value function $Q^{(h)}$ and the policy h. In general, this can only be achieved by storing distinct return estimates for every state-action pair. However, exact storage is not possible when the state variables have a very large or infinite number of possibles values. Therefore, ADP is used to do online reinforcement learning for solving the optimal control problem by using function approximation structures to estimate the value function. In fact, online LSPI[1,2] is an ADP algorithm that belongs to the approximate policy iteration (PI) techniques for ADP. In PI, the current policy is evaluated by computing its approximate value function which is then used to find a new improved policy.

3. PROBLEM FORMULATION

3.1 Assumptions

In this paper we focus on a set of robots, $\mathcal{R}$, made of two types of agents: L mobile sensors that form the set of sensors $\mathcal{S} = \{s_1, ..., s_L\}$ and one mobile router denoted by r. Therefore, $r \notin \mathcal{S}$ and $\mathcal{R} = \mathcal{S} \cup \{r\}$. We divide $\mathcal{R}$ in these two groups considering that the mobile sensors can be crawling robots, *i.e.*, OctoRoACHes[26], while the communication relays can be unmanned aerial vehicles, *i.e.*, quadrotors[27].

The set $\mathcal{R}$ operates in $\mathcal{W} \subset \mathbb{R}^3$ which is a compact subset of a three-dimensional Euclidean space where there is a fixed base station $\mathcal{b}$. In $\mathcal{R}$, there are M convex obstacles grouped in the set $\mathcal{O} = \{\mathcal{O}_1, ..., \mathcal{O}_M\}$ and there are N static rigid targets that forms the set $\mathcal{T} = \{\mathcal{T}_1, ..., \mathcal{T}_N\}$ such that $\mathcal{O} \cap \mathcal{T} = \emptyset$. We denote $I_{\mathcal{O}}$ and $I_{\mathcal{T}}$ as the

index sets of $\mathcal{O}$ and $\mathcal{T}$, respectively. For all $j \in I_{\mathcal{T}}$, we call $o_{\mathcal{T}j}$ to the center of $\mathcal{T}_j$. Meanwhile embedded in $\mathcal{W}$, there is a fixed Cartesian frame $\mathcal{F}_{\mathcal{W}}$ with origin $o_{\mathcal{W}}$. $\mathcal{F}_{\mathcal{W}}$ allows us to describe the position and orientation of the agents, objects and targets in $\mathcal{W}$. For instance, $\forall k \in I_O$, every point of $\mathcal{O}_k$ has a fixed position with respect to $\mathcal{F}_{\mathcal{W}}$ because the $\mathcal{O}_k$'s are considered rigid and fixed in $\mathcal{W}$ [16]. Figure 1(b) shows the 3D environment created in MATLAB for simulation purposes. Also, we include in this figure a legend on the right top corner to facilitate the reference of the different elements consider in this paper.

3.1.1 Motion Dynamics

Let $I_{\mathcal{S}}$ be the index set of $\mathcal{S}$, then $I_{\mathcal{S}} = \{1, ..., L\}$. We consider $\forall i \in I_{\mathcal{S}}$ s_i has a platform geometry $\mathcal{A}_i \subset \mathbb{R}^3$ and a field-of-view (FOV) geometry $\mathcal{V}_i \subset \mathbb{R}^3$ from which the robot can obtain sensor measurements[12,28], see Figure 1(b). We also assume $\mathcal{A}_i = \mathcal{A}_j$ and $\mathcal{V}_i = \mathcal{V}_j$ $\forall i, j \in I_{\mathcal{S}}$. Furthermore, $\mathcal{A}_i$ and $\mathcal{V}_i$ are both rigid and $\mathcal{V}_i$ has a fixed position and orientation with respect to $\mathcal{A}_i$. We say that $\forall i \in I_{\mathcal{S}}$ and $\forall j \in I_{\mathcal{T}}$ the sensor s_i gets measurements of the target $\mathcal{T}_j$ when $\mathcal{V}_i \cap \mathcal{T}_j \neq \emptyset$. In fact, we ensure this last condition when $s_i \in \mathcal{B}\left(o_{\mathcal{T}_j}, d_{o_{sens}}\right)$ where $d_{o_{sens}}$ is the minimum distance to the target $\mathcal{T}_j$ for getting measurements. We use $\mathcal{B}(\mathbf{q}, \delta)$ to denote the open ball of radius δ centered at $\mathbf{q}$. In addition, we suppose that $\mathcal{F}_{\mathcal{A}i}$ is a Cartesian frame embedded in $\mathcal{A}_i$ with origin $o_{\mathcal{A}i}$. Now let (x_{s_i}, y_{s_i}) be the position of $\mathcal{F}_{\mathcal{A}i}$ respect to $\mathcal{F}_{\mathcal{W}}$ and let θ_i be the orientation of $\mathcal{F}_{\mathcal{A}i}$ respect to $\mathcal{F}_{\mathcal{W}}$. Then, we define $\forall i \in I_{\mathcal{S}}$ the state vector of the sensor s_i as $\mathbf{q}_{s_i} = [x_{s_i}\ y_{s_i}\ \theta_i] \in SE(2)$. In other words, we consider that every s_i is moving just on the xy plane with θ_i as its heading angle. Notice that $\mathbf{q}_{s_i}$ can be used to determine the position and orientation of $\mathcal{A}_i$ and $\mathcal{V}_i$ respect to $\mathcal{F}_{\mathcal{W}}$. The state vector of each mobile sensor, $\mathbf{q}_{s_i}$, must also satisfy the sensor dynamics that are given by the unicycle model,

$$\begin{cases} \dot{x}_{s_i} = v_i \cos\theta_i, \\ \dot{y}_{s_i} = v_i \sin\theta_i, \\ \dot{v}_i = a_i, \\ \dot{\theta}_i = \omega_i, \end{cases} \tag{5}$$

where a_i and ω_i are the i^{th} mobile sensor's linear acceleration and angular velocity, respectively. Thus, the control vector for the s_i sensor is $\mathbf{u}_{s_i} = [a_i\ \omega_i] \in \mathbb{R}^2$.

On the other hand for the mobile router r, the range of communication coverage is denoted as δ_r. We also assume that r moves at a safe fixed height over the mobile sensors and over all the obstacles in $\mathcal{W}$. Furthermore, its motion dynamics are given by

$$\ddot{\mathbf{q}}_r = \mathbf{u}_r, \tag{6}$$

where $\mathbf{q}_r = [x_r\ y_r\ z_r] \in \mathbb{R}^3$ is the state vector of the mobile router and specifies its 3-D position respect to $\mathcal{F}_{\mathcal{W}}$, while $\mathbf{u}_r \in \mathbb{R}^3$ is its acceleration control input. We say that $\mathbf{u}_r = h(\mathbf{q}_r)$ where $h(\cdot)$ is the control policy for r.

3.1.2 Communication Links

For the next definitions, $\mathbf{q}^{(xy)} = \mathbf{q} \cdot \mathbf{v}_{xy}$ where $\mathbf{v}_{xy} = [1\ 1\ 0]^T$. In our scenario, see Figures 3(a) and 3(b), we assume there is a *point-to-point link* between the mobile router r and the base station $\mathcal{b}$ at any time. This link is represented by the magenta vector in Figures 3(a) and 3(b). Also, we suppose that r can manage at any time communication packets between any pair of sensors in $\mathcal{S}$ or between a sensor in $\mathcal{S}$ and $\mathcal{b}$. Indeed, we introduce the next two definitions.

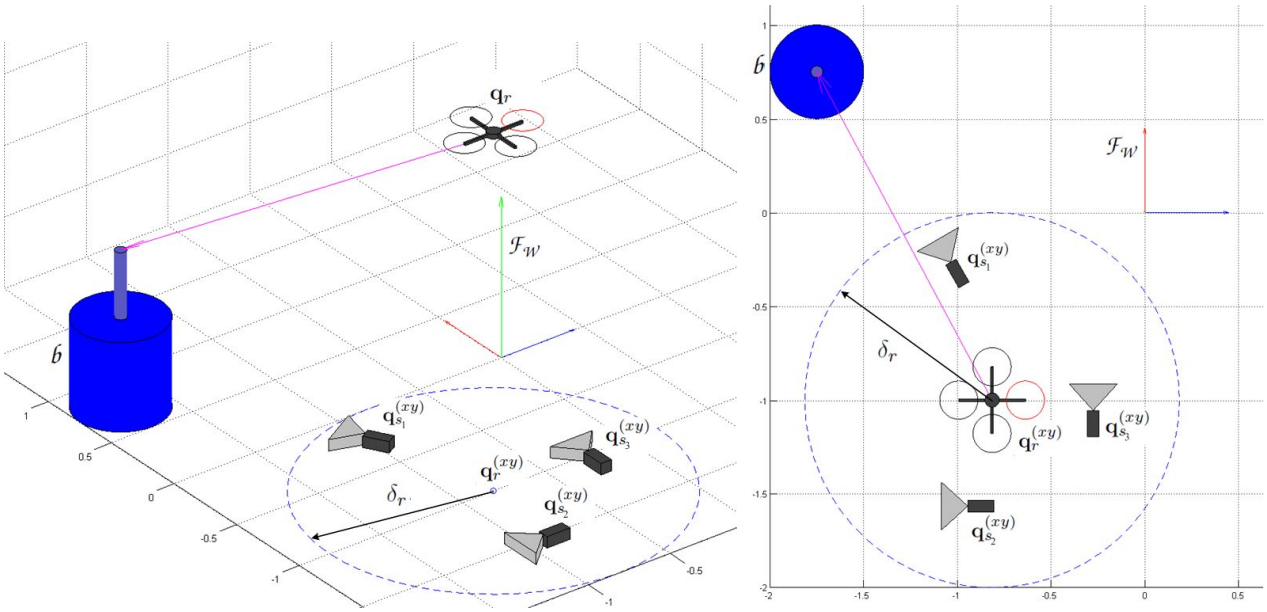
DEFINITION 3.1. *For all $i, j \in I_{\mathcal{S}}, i \neq j$, s_i has bidirectional communication with s_j if $\mathbf{q}_{s_i}^{(xy)}, \mathbf{q}_{s_j}^{(xy)} \in \mathcal{B}\left(\mathbf{q}_r^{(xy)}, \delta_r\right)$.*

DEFINITION 3.2. *For every $i \in I_{\mathcal{S}}$, s_i has bidirectional communication with $\mathcal{b}$ if $\mathbf{q}_{s_i}^{(xy)} \in \mathcal{B}\left(\mathbf{q}_r^{(xy)}, \delta_r\right)$.*

Therefore, a mobile sensors can talk with one of its pairs just if both are within the ball of radius δ_r and centered at $\mathbf{q}_r^{(xy)}$. Similarly, a mobile sensor can receive/send information from/to the base station just if it is within the ball of radius δ_r and centered at $\mathbf{q}_r^{(xy)}$. It is clear if Definition 3.2 holds for every $i \in I_S$ then Definition 3.1 also holds. Consequently, we can combine both definitions as next

DEFINITION 3.3. *If* $\forall i \in I_{\mathcal{S}}, \mathbf{q}_{s_i}^{(xy)} \in \mathcal{B}\left(\mathbf{q}_r^{(xy)}, \delta_r\right)$ *then* s_i *has bidirectional communication with any* s_j *where* $j \in I_{\mathcal{S}}, i \neq j$, *and* s_i *also has bidirectional communication with* $\mathcal{b}$.

Considering the explanation on Section 2.1 and from the assumptions detailed in Sections 3.1.1 and 3.1.2, $\mathcal{R}$ is an *heterogeneous robotic network* that exhibits hardware-based and objective-based heterogeneity. In fact, we are assuming agents with different dynamics and communication ranges, so $\mathcal{R}$ has hardware heterogeneity. Also, $\mathcal{R}$ presents objective-based heterogeneity because we assume that the ground agents objective is purely target sensing while the aerial agent objective is maintaining connectivity among the mobile sensors and the base station.



(a) (b)

Figure 3. Representation of the communication constraints specified by Definitios 3.1, 3.2 and 3.3. (a) 3D-view, and (b) 2D-view (xy view).

3.2 Problem Statement

Under the assumptions described in 3.1, we are concerned with the following problem: *A set of heterogeneous robots* $\mathcal{R}$ *formed by* L *mobile sensors and one mobile router* r *must obtain measurements of* M *targets located in an obstacle populated environment such that* $\mathcal{R}$ *maintains inter-agent connectivity and also connectivity with a base station* $\mathcal{b}$.

Since the set $\mathcal{R}$ works in a cluttered scenario, we have to add inter-robot collision prevention and obstacle avoidance to the objectives stated in the problem. Consequently, the problem considered in this paper aims to design a controller for the agents of the heterogeneous robotic network $\mathcal{R}$ such that they: (i) sense M static targets, (ii) keep inter-agent connectivity among the mobile sensors, (iii) keep mobile sensor and base station connectivity, (iv) avoid inter-agent collisions, and (v) avoid obstacle collisions. From the assumptions given in Section 3.1, the objectives (i) to (v) need to be considered for the design of the controller for the mobile sensors. On the contrary, the objectives (i), (iv) and (v) are not part of the controller design for the mobile router r because we assume that it has only communication and not sensing capabilities, and it always flies at a safe height over the mobile sensors and obstacles. In the next section, the proposed controllers for the mobile sensors and for the mobile router are described.

4. METHODOLOGY

From the problem statement, Section 3.2, we need to design two local controllers: one for the mobile sensors and one for the mobile router. Next, we present the methodologies around these controllers and then they will be tested in simulation, see Section 5. For the next definitions, $\|\cdot\|$ denotes the Euclidean norm.

4.1 Mobile Sensor Controller

In this case, the objectives (i) to (v) have to be accomplished by the controller of each $\mathcal{s}_i \in \mathcal{S}$. For the sensing objective (i), we consider that the set of mobile sensors $\mathcal{S}$ takes measurements of the M targets in $\mathcal{T} = \{\mathcal{T}_1, ..., \mathcal{T}_M\}$ in sequential order. This is first take measurements of $\mathcal{T}_1$ then of $\mathcal{T}_2$ and so on until $\mathcal{S}$ takes measurements of $\mathcal{T}_M$. For each $\mathcal{s}_i$, we design its controller based on potential field methods, Section 2.2, and specially taking as a based the potential functions given in equation (1), an attractive potential, and in equation (2), a repulsive potential. Thus, we define $U_i(\mathbf{q}_{s_i})$, the potential function of the ith mobile sensor with $i \in I_{\mathcal{S}}$ as

$$U_i(\mathbf{q}_{s_i}) = U_t(\mathbf{q}_{s_i}) + U_c(\mathbf{q}_{s_i}) + U_o(\mathbf{q}_{s_i}), \tag{7}$$

where $U_t(\mathbf{q}_{s_i})$ is the attractive potential of the jth target with $j \in I_{\mathcal{T}}$, $U_c(\mathbf{q}_{s_i})$ is a combination of an attractive potential for keeping inter-sensor connectivity and a repulsive potential for avoiding inter-sensor collision, and $U_o(\mathbf{q}_{s_i})$ is a repulsive potential for obstacle avoidance.

The attractive potential $U_t(\mathbf{q}_{s_i})$ is given by

$$U_t(\mathbf{q}_{s_i}) = \begin{cases} \frac{1}{2}\eta_t \varrho_t^2(\mathbf{q}_{s_i}, \mathcal{T}_j) & \text{if } \varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j) \geq d_{o_{sens}}, \\ 0 & \text{if } \varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j) < d_{o_{sens}}, \end{cases} \tag{8}$$

where $d_{o_{sens}}$ is the required distance from s_i to $\mathcal{T}_j$ to get sensor measurements, see Section 3.1.1, and $\varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j)$ is the distance between the mobile sensor s_i and the target $\mathcal{T}_j$ defined as

$$\varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j) = \max\left\{\|\mathbf{q}_{s_i}^{(xy)} - \mathbf{o}_1^{(xy)}\|, \|\mathbf{q}_{s_i}^{(xy)} - \mathbf{o}_2^{(xy)}\|\right\},$$

with $\mathbf{o}_1, \mathbf{o}_2 \in \mathcal{T}_j$, *i.e.*, $\mathbf{o}_1$ and $\mathbf{o}_2$ are the coordinates of two points within $\mathcal{T}_j$.

The potential $U_c(\mathbf{q}_{s_i})$ is defined as

$$U_c(\mathbf{q}_{s_i}) = \sum_{l=1, l\neq i}^{L} U(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}), \tag{9}$$

with

$$U(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) = \begin{cases} \frac{1}{2}\eta_c \varrho_s^2(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) & \text{if } \varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) > \frac{\delta_r}{2}, \\ \frac{1}{2}\eta_c \left(\frac{1}{\varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l})} - \frac{1}{d_{o_{col}}}\right)^2 & \text{if } \varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) \leq d_{o_{col}}, \\ 0 & \text{otherwise}, \end{cases} \tag{10}$$

where $\delta_r > d_{o_{col}}$, δ_r is the range of communication coverage, see Section 3.1.1, $d_{o_{col}}$ is the clearance inter-sensor distance, and $\varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) = \|\mathbf{q}_{s_i}^{(xy)} - \mathbf{q}_{s_l}^{(xy)}\|$.

The repulsive potential for obstacle avoidance $U_o(\mathbf{q}_{s_i})$ is given by

$$U_o(\mathbf{q}_{s_i}) = \sum_{k=1}^{M} U(\mathbf{q}_{s_i}, \mathcal{O}_k), \tag{11}$$

with

$$U(\mathbf{q}_{s_i}, \mathcal{O}_k) = \begin{cases} \frac{1}{2}\eta_{rep} \left(\frac{1}{\varrho_o(\mathbf{q}_{s_i}, \mathcal{O}_k)} - \frac{1}{d_{o_{obj}}}\right)^2 & \text{if } \varrho_o(\mathbf{q}_{s_i}) \leq d_{o_{obj}}, \\ 0 & \text{if } \varrho_o(\mathbf{q}_{s_i}) > d_{o_{obj}}, \end{cases} \tag{12}$$

where $\varrho_o(\mathbf{q}_{s_i}, \mathcal{O}_k)$ is the distance between s_i and the kth obstacle, and $d_{o_{obj}}$ is the influence distance of the obstacles.

Respect to the scaling factors η_t, η_c, and η_o, we assign their values according to Algorithm 1. Using this algorithm, we adapt the relative importance of the potential functions $U_t(\mathbf{q}_{s_i})$, $U(\mathbf{q}_{s_i}, \mathbf{q}_{s_l})$, and $U(\mathbf{q}_{s_i}, \mathcal{O}_k)$ between each other at every step time. For example, if a mobile sensor s_i is far from a target, but it is more possible a inter-sensor collision than an obstacle collision, *i.e.*, $e_c > e_o > e_t$, then we give more priority to the potential function $U(\mathbf{q}_{s_i}, \mathbf{q}_{s_l})$ with Algorithm 1 in order to avoid the inter-sensor collision.

Algorithm 1 Assignment of the scaling factors η_t, η_c and η_o

Input: : β_1, β_2, and β_3 positive scaling factors such that $\beta_1 > \beta_2 > \beta_3$

$e_t = \begin{cases} |\varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j) - d_{o_{sens}}| & \text{if } \varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j) \geq d_{o_{sens}} \\ 0 & \text{if } \varrho_t(\mathbf{q}_{s_i}, \mathcal{T}_j) < d_{o_{sens}} \end{cases}$

$e_c = \begin{cases} |\varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) - \frac{\delta_r}{2}| & \text{if } \varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) > \frac{\delta_r}{2} \\ |\varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) - d_{o_{col}}| & \text{if } \varrho_s(\mathbf{q}_{s_i}, \mathbf{q}_{s_l}) \leq d_{o_{col}} \\ 0 & \text{otherwise} \end{cases}$

$e_o = \begin{cases} |\varrho_o(\mathbf{q}_{s_i}) - d_{o_{obj}}| & \text{if } \varrho_o(\mathbf{x}) \leq d_{o_{obj}} \\ 0 & \text{if } \varrho_o(\mathbf{q}_{s_i}) > d_{o_{obj}} \end{cases}$

if $e_t \geq e_c$ **and** $e_t \geq e_o$ **then**
  $\eta_t = \beta_1$
  if $e_c \geq e_o$ **then**
    $\eta_c = \beta_2$, $\eta_o = \beta_3$
  else
    $\eta_c = \beta_3$, $\eta_o = \beta_2$
  end if
else if $e_t < e_c$ **and** $e_t \geq e_o$ **then**
  $\eta_t = \beta_2$, $\eta_c = \beta_1$, $\eta_o = \beta_3$
else if $e_t \geq e_c$ **and** $e_t < e_o$ **then**
  $\eta_t = \beta_2$, $\eta_c = \beta_3$, $\eta_o = \beta_1$
else
  $\eta_t = \beta_3$
  if $e_c \geq e_o$ **then**
    $\eta_c = \beta_1$, $\eta_o = \beta_2$
  else
    $\eta_c = \beta_2$, $\eta_o = \beta_1$
  end if
end if

The gradient of $U_i(\mathbf{q}_{s_i})$ is

$$\nabla U_i(\mathbf{q}_{s_i}) = \left[\frac{\partial U_i(\mathbf{q}_{s_i})}{\partial x_{s_i}} \; \frac{\partial U_i(\mathbf{q}_{s_i})}{\partial y_{s_i}} \; \frac{\partial U_i(\mathbf{q}_{s_i})}{\partial \theta_i} \right],$$

and then the artificial force induced by the potential function is $\mathbf{F}_i(\mathbf{q}_{s_i}) = -\nabla U_i(\mathbf{q}_{s_i})$. Now from Section 3.1.1, the control vector for the s_i sensor is $\mathbf{u}_{s_i} = [a_i \; \omega_i]$, so the potential-based control law[29] is given by

$$a_i = -[\cos\theta_i \; \sin\theta_i \; 0] \nabla U_i(\mathbf{q}_{s_i})^T - k_0 v_i, \tag{13}$$

$$\omega_i = k_1 \left[\text{atan2} \left(\frac{\partial U_i(\mathbf{q}_{s_i})}{\partial x_{s_i}}, \frac{\partial U_i(\mathbf{q}_{s_i})}{\partial y_{s_i}} \right) - \theta_i \right], \tag{14}$$

where k_0 and k_1 are positive constants.

4.2 Mobile Router Controller

The mobile router controller must satisfy the objectives (*ii*) and (*iii*) enumerated in Section 3.2. Indeed, these objectives are satisfied if Definition 3.3 holds. Thus, the mobile router r needs to maintain an adequate relative position respect to each one of the mobile sensors in $\mathcal{S}$. We formulate the next reward function for r in order to ensure its adequate position at all times,

$$\rho_c(\mathbf{q}_r) = \sum_{i=1}^{L} \zeta \left(e^{-\varrho_r(\mathbf{q}_r, \mathbf{q}_{s_i})/\tau} \right), \tag{15}$$

where ζ is a positive scaling parameter, $0 < \tau \ll 1$, and $\varrho_r(\mathbf{q}_r, \mathbf{q}_{s_i})$ is the separation distance given by

$$\varrho_r(\mathbf{q}_r, \mathbf{q}_{s_i}) = |\|\mathbf{q}_r^{(xy)} - \mathbf{q}_{s_i}^{(xy)}\| - \delta_r|. \tag{16}$$

In addition, a reward function $\rho_r(\mathbf{u}_r)$ is associated with the control input for the relay such that

$$\rho_r(\mathbf{u}_r) = \mathbf{u}_r^T \mathbf{R}_r \mathbf{u}_r, \tag{17}$$

where $\mathbf{R}_r = \text{diag}[a\ b\ c]$ with scalar weights $a,\ b,\ c > 0$. Considering the rewards (15) and (17), the total term function for the mobile router is

$$\Upsilon(\mathbf{q}_r, \mathbf{u}_r) = \rho_c(\mathbf{q}_r) - \rho_r(\mathbf{u}_r), \tag{18}$$

and using this function, we create the Q-function for the router as

$$Q^{(h)}(\mathbf{q}_r, \mathbf{u}_r) = \sum_{k=0}^{\infty} \gamma^k \Upsilon(\mathbf{q}_r(k), \mathbf{u}_r(k)), \tag{19}$$

where $\gamma \in [0, 1)$ is the discount factor. Therefore, our goal is to select the control policies $h(\mathbf{q}_r)$ such that $Q(\mathbf{q}_r, \mathbf{u}_r)$ is maximized.

Online LSPI algorithm[1,2] is used to solve this optimal control problem. This algorithm is an approximate PI technique for ADP (see Section 2.3). In general, the PI algorithm starts with an arbitrary initial policy h_0. Then at every iteration $\ell \geq 0$, it evaluates the current policy, *i.e.*, compute its Q-function Q^{h_ℓ} and finds an improved policy using

$$h_{\ell+1}(\mathbf{q}_r) = \arg\max_h \{Q^{(h_\ell)}(\mathbf{q}_r, \mathbf{u}_r)\} \tag{20}$$

The PI algorithm converges to an optimal policy h^*[23]. In online LSPI, the Q function is approximated using the linear parametrization

$$\hat{Q}(\mathbf{q}_r, \mathbf{u}_r) = \boldsymbol{\phi}^T(\mathbf{q}_r, \mathbf{u}_r)\boldsymbol{\theta} \tag{21}$$

where $\boldsymbol{\phi}(\mathbf{q}_r, \mathbf{u}_r) = [\phi_1(\mathbf{q}_r\ \mathbf{u}_r)\ ...\ \phi_m(\mathbf{q}_r, \mathbf{u}_r)]^T$ is a vector of m basis functions, BFs, and $\boldsymbol{\theta} \in \mathbb{R}^m$ is a parameter vector. In order to find the approximate Q-function of the current policy, online LSPI computes the parameter vector from a batch of samples running *least-squares temporal difference for Q-functions*, LSTD-Q,[24,30] that is a policy evaluation algorithm. Considering a set of samples $\left\{(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}}), \Upsilon(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}}) | l_s = 0, ..., n_s\right\}$ constructed by state-action samples and then computing next states and rewards, LSTD-Q process the samples using

$$\begin{aligned}
\boldsymbol{\Gamma}_{l_s+1} &= \boldsymbol{\Gamma}_{l_s} + \boldsymbol{\phi}(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}})\boldsymbol{\phi}^T(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}}) \\
\boldsymbol{\Lambda}_{l_s+1} &= \boldsymbol{\Lambda}_{l_s} + \boldsymbol{\phi}(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}})\boldsymbol{\phi}^T(\mathbf{q}_{r_{l_s+1}}, \mathbf{u}_{r_{l_s+1}}) \\
\boldsymbol{z}_{l_s+1} &= \boldsymbol{z}_{l_s} + \boldsymbol{\phi}(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}})\rho(\mathbf{q}_{r_{l_s}}, \mathbf{u}_{r_{l_s}})
\end{aligned} \tag{22}$$

with $\boldsymbol{\Gamma}_0 = 0, \boldsymbol{\Lambda}_0 = 0, \boldsymbol{z}_0 = 0$, and then solves the equation

$$\frac{1}{n_s}\boldsymbol{\Gamma}_{n_s}\hat{\boldsymbol{\theta}}^h = \gamma\frac{1}{n_s}\boldsymbol{\Lambda}_{n_s}\hat{\boldsymbol{\theta}}^h + \frac{1}{n_s}\mathbf{z}_{n_s} \tag{23}$$

to find an approximate parameter vector $\hat{\boldsymbol{\theta}}^h$. The solution $\hat{\boldsymbol{\theta}}^h$ found by LSTD-Q is substitute in (21) to get an approximate Q-function which is used to perform a policy improvement with (20) obtaining in this way an approximate PI algorithm. Indeed, online LSPI performs policy improvements once every few transitions before an evaluation of the current policy can be completed. Thus by *exploiting* the data collected by interaction, the online LSPI algorithm provides policy improvements. Furthermore, it *explores* other actions than those given by the current policy. In fact, the classical ε-greedy exploration[24] is used at every step k.

Algorithm 2 Online LSPI with ε-greedy exploration

Input: : discount factor γ, BFs $\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_n$, a small constant $\beta_{\boldsymbol{\Gamma}} > 0$
policy improvement interval $K_{\boldsymbol{\theta}}$, exploration schedule $\{\varepsilon_k\}_{k=0}^{\infty}$

$\ell = 0$, initialize policy h_0
$\boldsymbol{\Gamma}_0 = \beta_{\boldsymbol{\Gamma}} \mathbf{I}_{n\times n}, \boldsymbol{\Lambda}_0 = 0, \boldsymbol{z}_0 = 0$
measure initial state $\mathbf{x}_0$
for every time step $k = 0, 1, 2, ...$ **do**
  $\mathbf{u}_{r_k} = \begin{cases} h_\ell(\mathbf{q}_{r_k}) & \text{with probability } 1-\varepsilon_k \text{ (exploit)} \\ \text{a uniform random action} & \text{with probability } \varepsilon_k \text{ (explore)} \end{cases}$
  apply $\mathbf{u}_{r_k}$, measure next state $q_{r_{k+1}}$ and reward Υ_{k+1}
  $\boldsymbol{\Gamma}_{k+1} = \boldsymbol{\Gamma}_k + \boldsymbol{\phi}(\mathbf{q}_{r_k}, \mathbf{u}_{r_k})\boldsymbol{\phi}^T(\mathbf{q}_{r_k}, \mathbf{u}_{r_k})$
  $\boldsymbol{\Lambda}_{k+1} = \boldsymbol{\Lambda}_k + \boldsymbol{\phi}(\mathbf{q}_{r_k}, \mathbf{u}_{r_k})\boldsymbol{\phi}^T(\mathbf{q}_{r_{k+1}}, \mathbf{u}_{r_{k+1}})$
  $\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \boldsymbol{\phi}(\mathbf{q}_{r_k}, \mathbf{u}_{r_k})\rho(\mathbf{q}_{r_k}, \mathbf{u}_{r_k})$
  if $k = (\ell+1)K_\theta$ **then**
    solve $\frac{1}{k+1}\boldsymbol{\Gamma}_{k+1}\boldsymbol{\theta}_\ell = \gamma\frac{1}{k+1}\boldsymbol{\Lambda}_{k+1}\boldsymbol{\theta}_\ell + \frac{1}{k+1}\mathbf{z}_{k+1}$ ▷ finalize policy evaluation
    $h_{\ell+1}(\mathbf{x}) = \arg\max_h\{\boldsymbol{\phi}^T(\mathbf{q}_r, \mathbf{u}_r)\boldsymbol{\theta}_\ell\}$ ▷ policy improvement
    $\ell = \ell + 1$
  end if
end for

Algorithm 2 presents the online LSPI with ε-greedy exploration approach[1,2] applied for our case. The number $K_\theta \in \mathbb{N}, K_\theta \neq 0$ is the number of transitions between consecutive policy improvements. When $K_\theta = 1$ the policy is updated after every sample and the online LSPI is fully optimistic[23]. Meanwhile, the algorithm is partially optimistic when $K_\theta > 1$. In general, the number K_θ should not be chosen too large. About the exploration schedule $\{\varepsilon_k\}_{k=0}^{\infty}$, it should not approach to zero fast since a significant amount of exploration is recommended.

5. SIMULATION RESULTS

A 3-D environment is developed in MATLAB to test the methods proposed in Section 4. The sensor platform geometry as well as its FOV geometry are shown in Figure 1(b). On the other hand, the mobile routers are assumed as point robots, but they are represented as a thin cross with four circles at each side just for visualization purposes, see Figure 1(b) and Figure 4. Obstacle geometries and target geometries are assumed to be known *a priori*. The heterogeneous network considered for simulation is formed by 1 mobile router and 3 mobile sensors. The network moves in an environment $\mathcal{W}$ of $4\times4\times1.5$ m^3 with 4 obstacles and 2 targets, see Figure 4. The base station is located at the position $(-1.75, 1.75)$ m closed where the network starts moving to. The communication coverage radius for the mobile router is $\delta_r = 1$ m. The time dynamics, equation (5) for the sensors and equation (6) for the router, are implemented at each time step using ode45-differential equation solver in MATLAB. The step time is set up at 0.01 sec.

For the simulation, just the methodology proposed for the mobile sensors, Section 4.1, and Algorithm 1 are completely developed. We are currently working on the implementation of the sub-optimal controller for the mobile router. Therefore as initial test, we consider also a potential function control approach for the router where the potential function is given by

$$U_r(\mathbf{q}_r) = \sum_{i=1}^{L} U(\mathbf{q}_r, \mathbf{q}_{s_i}), \tag{24}$$

where

$$U(\mathbf{q}_r, \mathbf{q}_{s_i}) = \begin{cases} \frac{1}{2}\eta_r \varrho_{rs}^2(\mathbf{q}_r, \mathbf{q}_{s_i}) & \text{if } \varrho_{rs}(\mathbf{q}_r, \mathbf{q}_{s_i}) \geq \frac{\delta_r}{2}, \\ 0 & \text{if } \varrho_{rs}(\mathbf{q}_r, \mathbf{q}_{s_l}) < \frac{\delta_r}{2}, \end{cases} \tag{25}$$

with $\varrho_{rs}(\mathbf{q}_r, \mathbf{q}_{s_i}) = \|\mathbf{q}_r^{(xy)} - \mathbf{q}_{s_i}^{(xy)}\|$. Thus, the controller law is defined as

$$\mathbf{u}_r = -k_3 \begin{bmatrix} \frac{\partial U_r(\mathbf{q}_r)}{\partial x_r} & \frac{\partial U_r(\mathbf{q}_r)}{\partial y_r} & 0 \end{bmatrix}^T + k_4 \ddot{\mathbf{u}}_r, \tag{26}$$

where k_3 and k_4 are positive constants.

For Algorithm 1, we select $\beta_1 = 0.295, \beta_2 = 0.125,$ and $\beta_3 = 0.105$, and for the sensors control law, equations (13) and (14), we set $k_0 = k_1 = 0.75$. In the case of the mobile router controller, equation (26), $k_3 = 0.95$ and $k_4 = 0.85$ and in equation (25) $\eta_r = 0.425$. Figure 4 presents the 3-D view (parts (a) and (c)) and 2-D view (parts (b) and (d)) at two different time instants during the simulation. After 35 seconds, Figures 4(a) and 4(b), the heterogeneous network arrives to the first target and it is on its way to the second target at time equal to 40 seconds, Figures 4(c) and (d).

Even though we do not implement the methodology for the mobile router, Section 4.2, we calculate and plot at each time instant the proposed total reward term $\Upsilon(\mathbf{q}_r, \mathbf{u}_r)$ for the mobile router given in equation (18). We assume $\tau = 0.275$, $\eta = 0.5$, and $a = b = c = 0.5$. In fact, Figure 5(a) shows its plot. Also, the distance of each one of the sensors respect to the mobile router are presented in Figure 5(b). Around the time equal 38 seconds, the distances to the sensors grow and then the reward function decays drastically to a value of 5. On the contrary, the reward grows when the distances to the sensors decays as it can be checked at time equal around 5 seconds and around 30 seconds. Therefore, our proposed total reward term can be used as a based for the sub-optimal controller detailed in Algorithm 2.

6. CONCLUSION

This paper exploits the heterogeneity of a robotic network made of ground mobile sensors, *e.g.*, OctoRoACHes, and aerial mobile routers, *e.g.*, quadrotors, that is deployed to take measurements of multiple targets in a cluttered environment. Moreover, the platform and the field-of-view geometry just for the case of the ground sensors is taking in count in this work. Maintaining connectivity among the network members and a fixed base station is considered as part of the constraints for the design of the motion controller for the set of the network. Indeed, an adaptive potential field method is employed to coordinate the mobile sensors, while a sub-optimal controller based on the maximization of a reward function is formulated to move the mobile routers guaranteeing the inter-sensor communication coverage and the communication with the base station. Furthermore, a 3-D environment for simulation purposes has been created to test the potential field method proposed in this work. Future work will focus on implementing first in simulation the sub-optimal control algorithm for the mobile routers. Afterwards, an experimental verification of the proposed controllers will be implemented using the MARHES Lab heterogeneous robotic network.

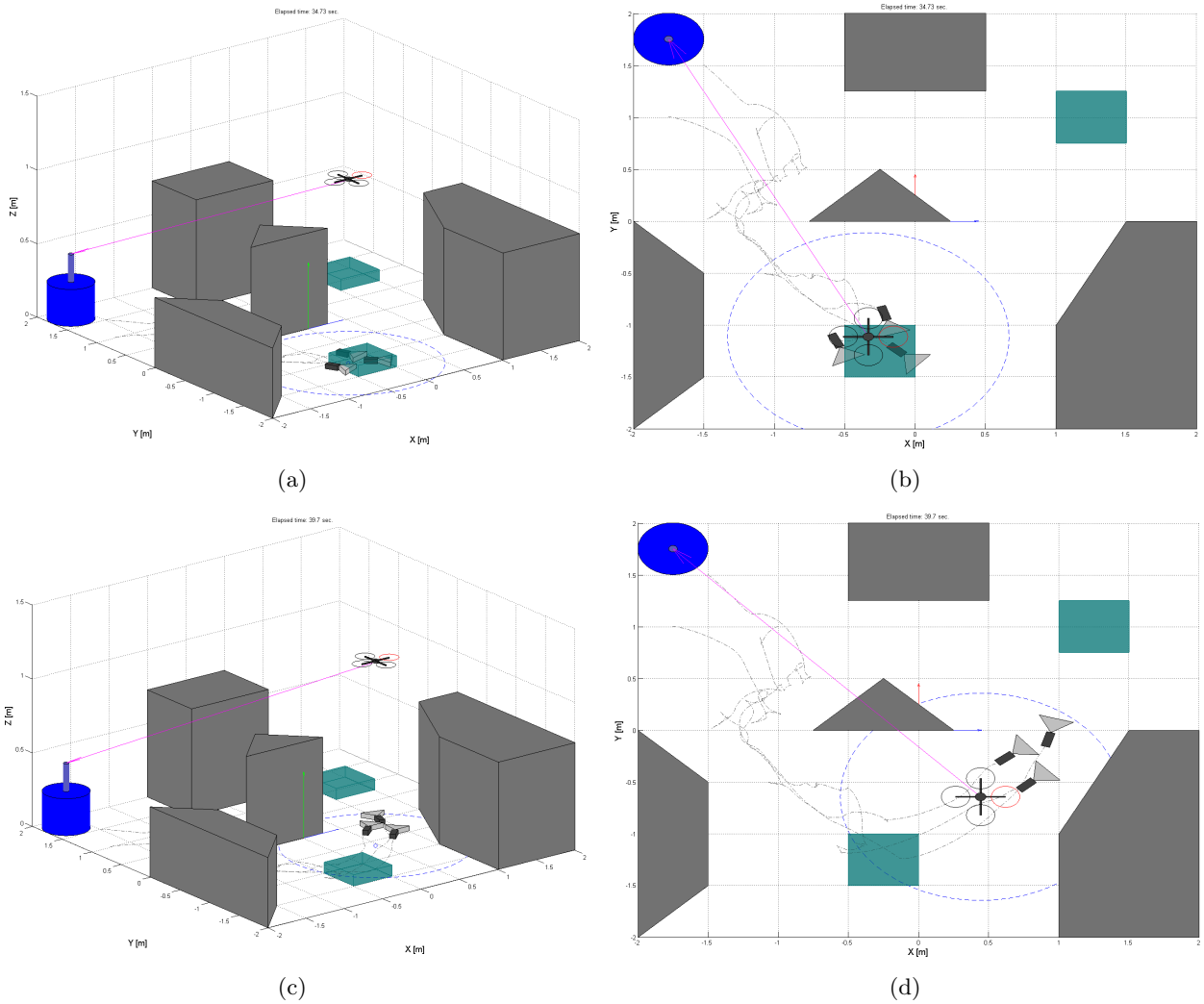
ACKNOWLEDGMENTS

This work was supported by NSF grants ECCS #1027775 and ECCS #1028506, and by the Army Research Laboratory grant #W911NF-08-2-0004. We would like to thank the SENESCYT Ecuadorian Scholarship Program for providing part of the financial support of P. Cruz.

REFERENCES

[1] Buşoniu, L., Babuška, R., Schutter, B. D., and Ernst, D., [*Reinforcement Learning and Dynamic Programming Using Function Approximators*], CRC Press, Inc., Boca Raton, FL, USA, 1st ed. (2010).

[2] Buşoniu, L., Ernst, D., De Schutter, B., and Babuška, R., "Online least-squares policy iteration for reinforcement learning control," in [*American Control Conference (ACC), 2010*], 486–491 (2010).



(a) (b)

(c) (d)

Figure 4. 3-D environment with 4 obstacles, 5 targets, 1 mobile router, and 3 mobile sensors. At two different time instants a 3-D view, (a) and (c), and a 2-D view, (b) and (d), are presented in this figure.

[3] Pereira, G. A. S., Das, A. K., Kumar, V., and Campos, M. F. M., "Decentralized motion planning for multiple robots subject to sensing and communication constraints," in [*Proceedings of the Second MultiRobot Systems Workshop*], 267–278, Kluwer Academic Press (2003).

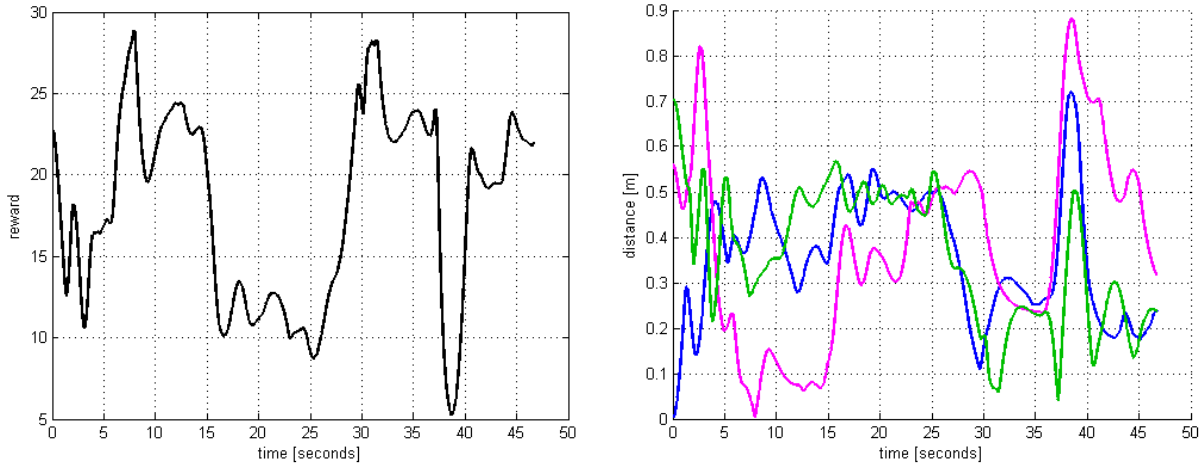
[4] Esposito, J. and Dunbar, T., "Maintaining wireless connectivity constraints for swarms in the presence of obstacles," in [*IEEE International Conference on Robotics and Automation (ICRA), 2006*], 946 – 951 (May 2006).

[5] Rooker, M. N. and Birk, A., "Multi-robot exploration under the constraints of wireless networking," *Control Engineering Practice* **15**(4), 435–445 (2007).

[6] Hsieh, M. A., Cowley, A., Kumar, V., and Taylor, C. J., "Maintaining network connectivity and performance in robot teams," *Journal of Field Robotics* **25**(1-2), 111–131 (2008).

[7] Fink, J., Ribeiro, A., and Kumar, V., "Motion planning for robust wireless networking," in [*IEEE International Conference on Robotics and Automation (ICRA), 2012*], 2419–2426 (2012).

[8] Gil, S., Schwager, M., Julian, B. J., and Rus, D., "Optimizing communication in air-ground robot networks using decentralized control," in [*IEEE International Conference on Robotics and Automation (ICRA), 2010*], 1964–1971 (2010).



(a) (b)

Figure 5. (a) The total term for the mobile router $\Upsilon(\mathbf{q}_r, \mathbf{u}_r)$, equation 18, and (b) the blue, magenta, and green plots correspond to the relative distance between the mobile router and the mobile sensor 1, 2, and 3, respectively.

[9] Bezzo, N., Anderson, M., Fierro, R., and Wood, J., "A real world coordination framework for connected heterogeneous robotic systems," in [*International Symposium on Distributed Autonomous Robotic Systems*], (November 2012).

[10] Cortez, R., Fierro, R., and Wood, J., "Connectivity maintenance of a heterogeneous sensor network," in [*Distributed Autonomous Robotic Systems*], Martinoli, A., Mondada, F., Correll, N., Mermoud, G., Egerstedt, M., Hsieh, M. A., Parker, L. E., and Sty, K., eds., *Springer Tracts in Advanced Robotics* **83**, 33–46, Springer Berlin Heidelberg (2013).

[11] Robuffo Giordano, P., Franchi, A., Secchi, C., and Bülthoff, H. H., "A passivity-based decentralized strategy for generalized connectivity maintenance," *The International Journal of Robotics Research* **32**(3), 299–323 (2013).

[12] Zhang, G. and Ferrari, S., "An adaptive artificial potential function approach for geometric sensing," in [*Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference (CDC/CCC 2009)*], 7903 –7910 (December 2009).

[13] Pimenta, L. C. A., Kumar, V., Mesquita, R., and Pereira, G. A. S., "Sensing and coverage for a network of heterogeneous robots," in [*47th IEEE Conference on Decision and Control, 2008. CDC 2008.*], 3947–3952 (2008).

[14] Iocchi, L., Nardi, D., Piaggio, M., and Sgorbissa, A., "Distributed coordination in heterogeneous multi-robot systems," *Autonomous Robots* **15**(2), 155–168 (2003).

[15] Bezzo, N., Griffin, B., Cruz, P., Donahue, J., Fierro, R., and Wood, J., "A cooperative heterogeneous mobile wireless mechatronic system," *IEEE/ASME Transactions on Mechatronics* **PP**(99), 1–12 (2012).

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

[17] Ge, S. and Cui, Y., "New potential functions for mobile robot path planning," *IEEE Transactions on Robotics and Automation,* **16**(5), 615–620 (2000).

[18] Zou, X.-y. and Zhu, J., "Virtual local target method for avoiding local minimum in potential field based robot navigation," *Journal of Zhejiang University SCIENCE A* **4**(3), 264–269 (2003).

[19] Chengqing, L., Ang, V., Krishnan, H., and Lim, S. Y., "Virtual obstacle concept for local-minimum-recovery in potential-field based navigation," in [*IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA '00.*], **2**, 983–988 vol.2 (2000).

[20] Lewis, F. and Liu, D., eds., [*Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*], IEEE Press Series on Computational Intelligence, Wiley (2012).

[21] Lewis, F. and Vrabie, D., "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine* **9**(3), 32–50 (2009).

[22] Si, J., Barto, A. G., Powell, W. B., and Wunsch, D., [*Handbook of Learning and Approximate Dynamic Programming*], IEEE Press Series on Computational Intelligence, Wiley-IEEE Press (2004).

[23] Bertsekas, D. P., [*Dynamic Programming and Optimal Control, Vol. I and Vol. II*], Athena Scientific, 3rd (Vol. I) and 4th (Vol. II) ed. (2005 (Vol. I) and 2012 (Vol. II)).

[24] Powell, W. B., [*Approximate Dynamic Programming: Solving the Curses of Dimensionality*], Wiley Series in Probability and Statistics, Wiley-Interscience (2007).

[25] Wang, F.-Y., Zhang, H., and Liu, D., "Adaptive dynamic programming: An introduction," *IEEE Computational Intelligence Magazine* **4**(2), 39–47 (2009).

[26] Pullin, A., Kohut, N., Zarrouk, D., and Fearing, R., "Dynamic turning of 13 cm robot comparing tail and differential drive," in [*2012 IEEE International Conference on Robotics and Automation (ICRA)*], 5086 –5093 (May 2012).

[27] Bouabdallah, S. and Siegwart, R., *Design and control of quadrotors with application to autonomous flying*, Thèse sciences, Faculté des sciences et techniques de l'ingénieur STI, Section de microtechnique, Institut d'ingénierie des systèmes I2S (Laboratoire de systèmes autonomes 1 LSA1). Dir.: Roland Siegwart, Ecole polytechnique fédérale de Lausanne, EPFL, Lausanne (2007).

[28] Ferrari, S., Anderson, M., Fierro, R., and Lu, W., "Cooperative navigation for heterogeneous autonomous vehicles via approximate dynamic programming," in [*2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*], 121–127 (2011).

[29] Ferrari, S., Fierro, R., and Wettergren, T., [*Modeling and Control of Dynamic Sensor Networks*], CRC Press, Inc. (2013). (To be published).

[30] Lagoudakis, M. G., Parr, R., and Littman, M. L., "Least-squares methods in reinforcement learning for control," in [*SETN 02: Proceedings of the Second Hellenic Conference on AI*], 249–260, Springer-Verlag (2002).