2-8-2011

Prioritized Sensor Detection with Communication Constraints: A Cyber-Physical Systems Approach

Randy Andres Cortez

Prioritized Sensor Detection with Communication Constraints: A Cyber-Physical Systems Approach

by

Randy Andres Cortez

B.S., Mathematics, New Mexico Highlands University, 2005
M.S., Mechanical Engineering, University of New Mexico, 2007

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctorate of Philosophy
Engineering

The University of New Mexico

Albuquerque, New Mexico

December, 2010

Dedication

To my family and friends for their support, encouragement, and patience.

"Focus on the journey, not the destination. Joy is found not in finishing an activity but in doing it." – Greg Anderson

Acknowledgments

This work is supported by the Department of Energy URPR Grant: DE-FG52-04NA255590.

I would like to thank my advisor, professor Rafael Fierro, who gave me the opportunity to carry out this research, for his kind support, and his believing in me and my work. I would also like to thank professor Herbert Tanner, professor Ron Lumia, and professor Greg Starr for serving as members on my committee.

Lastly I would like to thank my parents, Joseph and Babbie, my two sisters, Mandy and Dee Dee, and especially Angela for their support and motivation.

Prioritized Sensor Detection with Communication Constraints: A Cyber-Physical Systems Approach

by

Randy Andres Cortez

ABSTRACT OF DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctorate of Philosophy
Engineering

The University of New Mexico

Albuquerque, New Mexico

December, 2010

Prioritized Sensor Detection with Communication Constraints: A Cyber-Physical Systems Approach

by

Randy Andres Cortez

B.S., Mathematics, New Mexico Highlands University, 2005
M.S., Mechanical Engineering, University of New Mexico, 2007

Ph.D., Mechanical Engineering, University of New Mexico, 2010

Abstract

Currently in the literature there does not exist a framework which incorporates a heterogeneous team of agents to solve the sensor network connectivity problem. An approach that makes use of a heterogeneous team of agents has several advantages when cost, integration of capabilities, or possible large search areas need to be investigated. A heterogeneous team allows for the robots to become "specialized" in their abilities and therefore accomplish sub-goals more efficiently which in turn makes the overall mission more efficient.

In Part I of this dissertation we address the problem of prioritized sensing of an area with a homogeneous sensor network. We derive a decentralized and collision free controller that drives the sensing agents to positions within the area that contain the highest probability of containing “good information.” We then apply this prioritized sensing controller to a target search scenario, where a group of cooperating UAVs must detect then track a maneuvering target within the search space.

In Part II of this dissertation we relax the assumption of network connectivity within the sensor network and introduce mobile communication relays to the network. This addition converts the homogeneous sensor network to a heterogeneous one. Based on the communication geometry of both sensing and communication relay agents we derive communication constraints within the network that guarantee network connectivity. We then define a heterogeneous proximity graph that encodes the communication links that exist within the heterogeneous network. By specifying particular edge weights in the proximity graph, we provide a technique for biasing particular connections within the heterogenous sensor network. Through a minimal spanning tree approach, we show how to minimize communication links within the network which allows for larger feasible motion sets of the sensing agents that guarantee the network remains connected. We also provide an algorithm that allows for adding communication links to the minimal spanning tree of the heterogeneous proximity graph to create a biconnected graph that is robust to a single node failure. We then combine the prioritized search algorithm from Part I and the communication constraints from Part II to provide a decentralized prioritized sensing control algorithm for a heterogenous sensor network that maintains network connectivity.

Lastly, in Part III we describe our robotic testbed that has been built to validate our proposed algorithms. We provide hardware experiments for both homogenous and heterogeneous sensor networks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we list some motivating applications that have lead to our current problem formulation. A literature review of related work in the field is given and an overview of our approach is explained. Lastly we present a short description of the contributions of this work.

1.1 Motivation

Robotic motion planning is a well-addressed issue in autonomous systems, [1]. However a growing number of applications such as spatial distribution mapping, dynamic sensing coverage, and dynamic target detection, have motivated navigation and control algorithms for teams of goal-oriented mobile sensor networks. When considering control and coordination algorithms of reconfigurable sensor networks one must join the coordination/navigation of the robots with the sensing cost or desired configurations of the sensor network. In problems involving reconfigurable sensor networks, a primary goal is to reconfigure the sensor network in such a way that the time taken to reconfigure is minimized or the sensing coverage is maximized. This has useful applications in target detection and surveillance as well as spatial distribution mapping, among many others.

Recently in the literature, connectivity maintenance has been considered as a constraint on the reconfigurable sensor network. The constraint of maintaining connectivity between sensor nodes is a relaxation to the typically assumed fixed communication topology. The connectivity constraint complicates the motion planning problem for the reconfigurable sensor network in the sense that sensors should only move to areas in the search space where communication can be guaranteed. Typically the connectivity constraint is directly imposed on the sensor network, which may greatly limit the sensor networks ability to investigate the search space. To overcome this constraint on the reconfigurable sensor network we propose to add "relay" agents to the communication topology. This allows for the reconfigurable sensor network to have "a longer reach" to investigate the search space, with the added cost of having to control a heterogeneous team of robots (sensors and relays).

In general, approaches to reconfigurable sensor networks use gradient type algorithms to reconfigure the network for sensing optimality [2]. Using a gradient type approach has been shown to be successful when the underlying density function is static and the environment is free of obstacles. These local gradient type approaches which address sensing coverage problems cannot however, address certain types of reconfigurable sensor network problems. One very interesting problem, which we consider here, is that of biasing certain regions of the area-of-interest to be searched because of prior knowledge or because an underlying time constraint that prohibits an exhaustive search of the area, *i.e.,* emergency situations, search and rescue. Gradient type approaches may fail to send the robots to places that have the highest likelihood of containing the most useful information in the time allotted because of the local nature of the gradient technique. Also the probability of certain regions containing useful information will be changing as the robots search the area, which makes the underlying density function in this scenario dynamic. The problem here becomes, how to coordinate the sensor-enabled robot team in such a way that areas within the region that have the most probability of containing "good" information are searched first while also maintaining a connected communication network. The label "good" here may mean a target of interest, hazardous material, a group of people to be rescued, etc., depending on the particular application the robot team is tasked with. Because of prior knowledge, or due to time constraints, the robots should search the most likely areas of finding "good" information first and continue until enough information is gathered or the time constraint is met. The scenario stated here is quite different from other sensor network problems because the reconfigurable sensor network is not trying to achieve optimal sensing coverage over the area-of-interest nor exhaustively search the space, rather the robot team is trying to search the most probable places of containing "good" information first because the search is biased by prior knowledge or because of a time constraint may greatly limit the robot team's ability to search the entire area of interest.

Some motivating and practical applications include search and rescue operations [3], target detection [4], and hazardous contaminations [5] to name a few. In these scenarios, regions within a given area that are most likely to contain humans, enemy targets, or hazardous material, should be searched first while regions with less probability of containing these features are searched later. As an added difficulty to the prioritized search, we also require the communication topology to be connected at each new network configuration.

1.2 Prior Work

1.2.1 Sensor Networks

Recent research in sensor networks concentrates on creating a sensor network that can adapt to its environment. In this sense, the trend is towards reconfigurable sensor networks. Addressing reconfigurable sensor networks is typically done by utilizing mobile sensor platforms to adapt to the environment, [6] and [7]. Mobile robots give the network the ability to react to changes in the environment through their mobility by placing sensors to more interesting areas or places that may lack sensor coverage due to spatial configurations or possible sensor failures. Mobile robots also allow the network to verify or disregard abnormal (noisy) data coming from a sensor by reconfiguring so that a second sensor can obtain data to compare with.

Much of the research in reconfigurable sensor networks focuses on surveillance and tracking tasks. The goal in these applications is to identify and track targets moving within the area the sensor network covers. In [6], Huntwork *et al.*, create a sensor network with both mobile sensor platforms as well as stationary sensors. The network is not assumed to be configured in any optimized fashion, so mobile sensors are used when a target is no longer visible by the stationary sensors, or cannot be tracked at an adequate resolution. In [7], Singh *et al.*, propose to reconfigure a mobile sensor network through an active learning technique for the purpose of mapping lake water current velocities to study hydrodynamic effects. Active learning assumes that initially mobile sensor platforms are uniformly distributed in the area of investigation. The mobile sensors then do a coarse survey of the area to obtain a rough estimate of the features in the environment. Subsequent more refined passes over the area are done over paths that are planned based on data from previous passes. In [8], Oh *et al.*, develop an efficient multi-sensor fusion algorithm that utilizes binary target detection data from a sensor network to track the trajectory of the target by using spatial correlation.

Cortés, Martínez, and Bullo [2] and [9], use gradient climbing algorithms to distribute sensor platforms in an optimal fashion over the area in question to address spatial distribution of the sensor platforms. Robot agents follow gradients that maximize a static density function that is weighted by a sensor performance function. The area is divided up among the agents using Voronoi partitions. Hussein and Stipanovic [10] use a gradient climbing method for control of the sensor network as well, but without having to partition the area among the team members, which reduces computational overhead.

A large part of research in reconfigurable sensor networks is mostly concerned with issues related to monitoring, surveillance, and target tracking, but there are also applications beyond pure surveillance or tracking [7]. Mobile sensor platforms can be used as reconfigurable sensor networks for mapping spatial distributions of physical quantities [11]. We envision controlling a multi-robot team to map these physical quantities, where the sensor network is reconfigured on-line based on the information from the sensed environment.

1.2.2 Heterogeneous Multi-Robot Control

Research in the area of heterogeneous multi-robot control is beginning to receive attention from the research community. Pimenta *et al.* [12] address the problem of covering an environment with robots equipped with sensors which are heterogeneous in the sense that the sensor footprints are different. They achieve optimal distributed control laws by utilizing the locational optimization framework coupled with constraints. In a very similar approach, Lam and Liu [13] develop two algorithms for mobile agent deployment for sensor coverage enhancement. The mobile agent team is also seen as heterogeneous through different sensing radii, however the algorithms developed rely on a circle packing technique as well as an artificial potential approach. Another approach that looks at heterogeneity of the network through differing sensor footprints can be found in [14]. In [15], Tanner and Christodoulakis address the problem of coordinating UGVs and UAVs for cooperative intelligence, surveillance, and reconnaissance missions. The controllers for both groups are achieved through a swarm control approach. Lyapunov analysis establishes stability of the ground vehicle motion and also insures the tracking performance of the aerial vehicles. In [16], Kumar *et al.* derive a decentralized self sorting method based on artificial potentials to segregate heterogeneous robot agents. These segregation methods may be useful in deployment and organization of heterogeneous robot teams. Parker *et al.* [17], [18] presents autonomous behaviors for tightly-coupled cooperation in heterogeneous robot teams, specifically for navigation assistance. These behaviors enable "leader" robots to assist in the navigation of "simple" robots that have no onboard capabilities of obstacle avoidance.

1.2.3 Multi-Robot Coordination with Connectivity

Recently in the literature, connectivity maintenance has been considered as a constraint on the reconfigurable sensor network. The constraint of maintaining connectivity between sensor nodes is a relaxation to the typically assumed fixed communication topology. The connectivity constraint complicates the motion planning problem for the reconfigurable sensor network in the sense that sensors should only move to areas in the search space where communication can be guaranteed. Typically the connectivity constraint is directly imposed on the sensor network, which may greatly limit its ability to investigate the search space. To overcome this constraint on the reconfigurable sensor network we propose to add mobile communication "relay" agents to the communication topology. This allows the reconfigurable sensor network to have a "longer reach" to investigate the search space, with the added difficulty of having to control a heterogeneous team of robots (sensors and relays).

Research in multi-robot coordination typically assumes that the underlying communication topology is fixed and connected, [19], [20]. Recently however, research has begun to focus on a relaxation of this assumption, namely considering the connectivity of the multi-robot group as being a dynamic topology which should maintain some connectivity properties. In [21], Dimarogonas and Johansson present a distributed control law that guarantees connectivity maintenance in a network of multiple mobile agents. The control law is achieved through a potential field approach with guaranteed boundedness on the agents input. Muhammad *et al.,* [22] derive graph processes to pre-plan for formation tasks, taking into account the graph connectivity. In [23], abstractions are used to enable multiple groups of agents to form desired formations when communication between these groups is limited due to high bandwidth cost. Michael *et al.,* [24] implement a control algorithm that is based on a consensus approach and market based auctions [25] on a group of seven mobile robots. The local connectivity of the group is estimated by computing the second smallest eigenvalue of the graph Laplacian, similar to the work in Kim and Mesbahi [26]. In [27], Ji and Egerstedt address maintaining connectivity in rendezvous and formation control problems. Spanos and Murray [28] derive a function that measures local connectedness of the communication network. This function also provides a sufficient condition for global connectedness of the communication network. Fink and Kumar [29] explore methods for online mapping of Received Signal Strength Indicator (RSSI) with mobile robots where the RSSI map can then be used for control algorithms requiring inter-robot communications. Tekdas *et al.,* [30], [31] study the problem of computing the minimum number of robotic routers in order to maintain connectivity of a single user to a base station. A similar approach is taken by Burdakov *et al.,* [32], where an array of possible relay chains are computed and a user is allowed to choose the one that best fits their needs in terms of cost and number of communication hops.

Works more closely related to the work in this paper can be seen in the following. Reference [33] develops a distributed controller to position a team of UAVs in a configuration that optimizes communication-link quality to support a team of UGVs performing a collaborative task, however the authors must assume the UGVs do not move to guarantee the connectivity of the combined UAV, UGV network. Reference [34] introduces the idea of periodic connectivity where the network must regain connectivity at a fixed interval. The authors propose an implicit coordination algorithm that allows all robots to plan assuming all other robots are stationary, then plans are exchanged to improve performance. It is unclear how many communication rounds are needed to reach a global consensus plan or if this type of algorithm will end up in a deadlock configuration. Reference [35] presents an algorithm that allows a robot to determine when it is feasible for it to move to a desired point by adjusting its own position while maintaining network connectivity. This is achieved by solving a convex optimization problem in an incremental fashion, however the extension to multiple robots moving to multiple desired points is not addressed. In [36], Frew presents an information-theoretic framework to integrate sensing and communication for planning of robot sensor networks. This approach takes uncertainty into account, however it is not clear if any type of connectivity guarantees can be made about the network. Also only static sensing scenarios can currently be addressed in this framework. In [37] the authors derive a flocking controller to regulate the distance between vehicles that address coverage and vehicles that address coordination. The distance requirements for the flocking controller are the communication range of the vehicle types. Stachura and Frew [38] use a fixed planning hierarchy for a finite horizon optimization that addresses cooperative target localization with communication considerations. To determine trajectories for the sensor network the first sensor plans its trajectory over the time horizon then subsequent sensors plan based on the trajectories of the sensors that are higher in the hierarchy.

1.3 Problem Statement

Currently in the literature there does not exist a framework which incorporates a heterogeneous team of agents to solve the sensor network connectivity problem in search and sensing scenarios. An approach that makes use of a heterogeneous team of agents has several advantages when cost, integration of capabilities, or possible large search areas need to be investigated. A heterogeneous team allows for the robots to become "specialized" in their abilities and therefore accomplish sub-goals more efficiently which in turn makes the overall mission more efficient.

Our goal is to develop a framework that can guarantee connectivity in a group of heterogeneous agents whose mission objective is a prioritized search of an area. Such a framework would help to overcome the limitations imposed by a homogeneous team of agents trying to accomplish the same mission.

1.4 Overview of Proposed Approach

A cyber-physical system (CPS) is a network of physically distributed sensors and actuators capable of computation, communication, and control that relies highly on the integration of these capabilities for its operation and interaction with the physical environment in which it is deployed. Our solution, which involves the development of a framework for a heterogeneous sensor network, can been viewed as a cyber-physical system.

We envision a CPS to address this problem as a group of autonomous agents, possibly both ground and aerial vehicles, that are equipped with environmental sensing capabilities, a communication network, as well as having the capability of receiving control inputs. Through the interaction and coordination of the autonomous agents, a connected network topology can be maintained which is critical to accomplish-

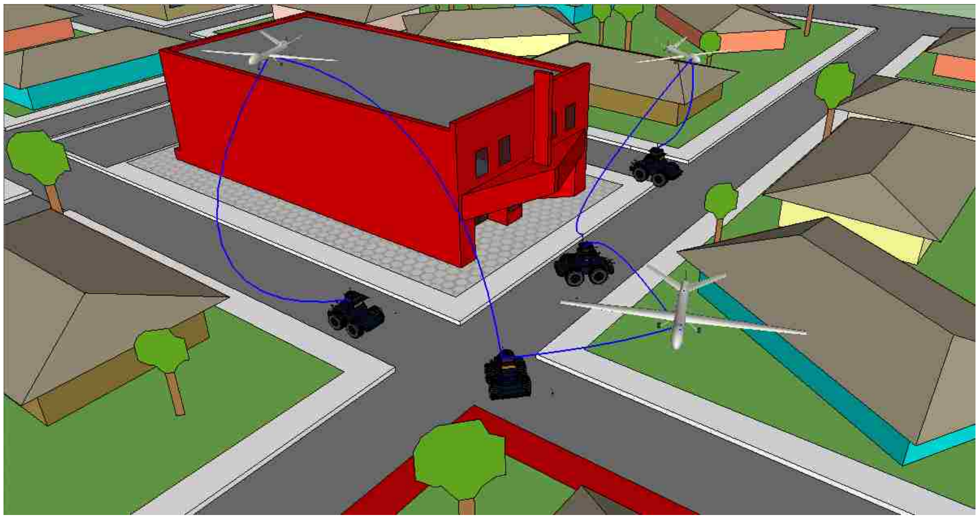

Figure 1.1: Envisioned CPS with both ground and aerial vehicles used for obtaining situational awareness in an emergency situation. Communication links between aerial and ground vehicles enables coordination as well as the ability to relay real-time environmental information to an end-user.

ing the mission objective. A visual representation of such a CPS can be seen in Figure 1.1.

In our approach, we begin by developing a prioritized search algorithm for a homogeneous sensor network where we assume network connectivity is guaranteed over the search area. We then relax the assumption of network connectivity and introduce specialized mobile "relay" agents to the network which are better equipped to communicate over longer distances. We develop communication constraints for the newly formed heterogeneous sensor network comprised of both sensing and relay agents. These constraints lead to feasible motion sets for each sensing agent that can be optimized with respect to the prioritized search and still guarantee network connectivity. Therefore, our approach of controlling a heterogeneous sensor network allows for the unification of prioritized search over an area while maintaining network connectivity.

1.5 Contributions of Dissertation

Although recent research has begun to address problems in heterogeneous multi-robot control, it has been limited largely to problems where the sensing foot print is different among the agents, or when the underlying vehicle dynamics differ between agents. We propose to address the problem of controlling heterogeneous agents within a team that have different objectives and "abilities" but contribute to an overall mission. This differs from other works in the literature in the sense that we are addressing the "coupling" of subgoals to achieve an overall global goal. Also, our approach to connectivity maintenance does not rely on the algebraic connectivity of the underlying graph topology like much of the current literature. Through a geometric approach we are able to specify feasible motion sets that guarantee network connectivity maintenance.

The problem addressed in this paper is slightly different from others in the literature since we are not trying to achieve a flocking or swarming behavior nor are we interested in controlling formations while maintaining connectivity. These problems have a similar theme, agents should be close to each other for the objective to be accomplished. Sensing or search problems on the other hand, which we are addressing here, need the agents to "spread out" in order for the search algorithm to be effective. This allows the agents to search in a parallel fashion. With this in mind, our approach to multi-robot connectivity maintenance we propose adding specialized agents that are better equipped (hardware) to relay information over longer distances to the sensor network. This allows the sensor network to have a longer "reach" in the search space. It also allows the sensing agents to be built in a way that the communication hardware can be minimized. This may have benefits in hardware costs and also the size and shape of the sensing agents.

To this end, we first develop an algorithm for a prioritized search of an area with a homogeneous sensor network. Our prioritized search algorithm guarantees inter-robot collision avoidance as well as collision avoidance with general shaped obstacles in the environment. The prioritized sensing controller is then applied to a target search scenario, where a group of cooperating UAVs must detect then track a maneuvering target within the search space. We then develop heterogeneous communication constraints that describe under what conditions network connectivity can be maintained. With these communication constraints in mind, we develop a heterogenous proximity graph that describes the communication connections (links) for a heterogeneous sensor network made up of sensing and relay agents. We then derive edge weights for the communication graph that allows for shaping the flow of information within the network and then show how to minimize the communication constraints of the network. The minimization of communication constraints on the sensor network leads to a larger feasible search area for the sensing agents. We then show how these communication constraints lead to a feasible motion set for each agent in the network that guarantees connectivity. To address robustness of the network to node failures, we provide an algorithm to add communication links to the minimized network topology to create a biconnected graph. The biconnected graph allows the network to remain connected in the case of a single node failure. Lastly, we combine the prioritized search with the feasible motion sets to provide an algorithm that can conduct a prioritized search of an area with maintaining network connectivity. We also describe how the prioritized search algorithm with connectivity constraints can be implemented in a decentralized fashion.

1.6 Organization of Dissertation

Part I consists of a first approach to reconfigurable sensor networks using a homogeneous sensor network. Particularly, in Chapter 2 we review some technical background used throughout the dissertation. In Chapter 3 we provide details on how to do a prioritized search with a homogeneous sensor network. We also provide a technique to address situations when robot team members have multiple sensing capabilities. Chapter 4 shows the adaptability of our prioritized search algorithm by applying it to a search and tracking problem for maneuvering targets of interest.

Part II extends our approach to reconfigurable sensor networks by introducing specialized agents to the network. These specialized agents are introduced to overcome relaxations to network connectivity. This shifts the focus of the approach from controlling a homogeneous sensor network to controlling a heterogeneous sensor network. In chapter 5 we develop communication constraints for the heterogeneous sensor network that guarantees the network stays connected. Chapter 6 combines the heterogeneous communication constraints with our prioritized search algorithm.

Part III looks at experimental results for both the homogeneous and heterogeneous sensor networks. Specifically, chapter 7 describes the experimental testbed used for our qualitative validation of the proposed algorithms. Chapter 8 describes results of the prioritized sensing algorithm for a homogeneous sensor network. Chapter 9 details experimental results of the heterogeneous communication constraints. Lastly, chapter 10 gives some concluding remarks and thoughts on future work directions.

Part I: Homogeneous Sensor Network

Chapter 2

Technical Background

In this chapter we review some technical background relevant to our approach. Specifically, Voronoi partitioning, randomized global optimization, mobile robot motion planning, and graph connectivity considerations. This technical background lays down the foundation for our mathematical problem formulation.

2.1 Task Decomposition

We define task decomposition as "dividing" or partitioning the overall task among the available autonomous agents. A natural way of accomplishing this decomposition is the notion of Voronoi partitioning where the centroid of each Voronoi cell is taken to be the position of a single mobile robot. Thus, a certain region within this area (namely the corresponding Voronoi cell) is allocated to each robot for searching. This is performed on an iterative basis, so the Voronoi partitions are dynamic in nature.

Using Voronoi partitions, the area to be mapped can be broken up dynamically among the robot team members based on their current locations. Also by construction, Voronoi partitioning can be implemented in a decentralized fashion.

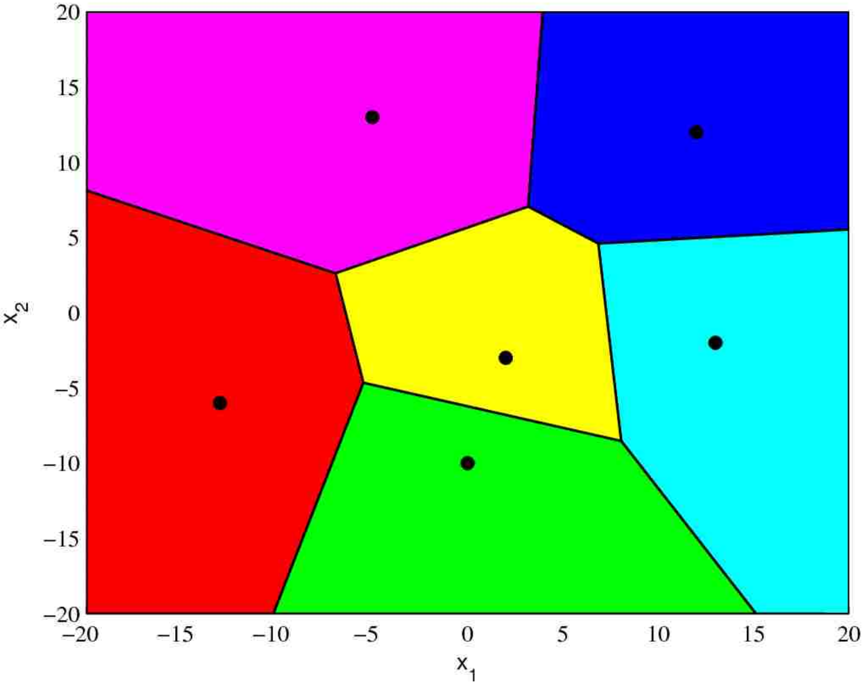

Figure 2.1: Voronoi partitioning of a search space with six agents in $\mathbb{R}^2$. The different colors represent each agents corresponding Voronoi Cell.

Our area to be searched, Q, is assumed to be a simple convex polygon in $\mathbb{R}^N$ including its interior. Let P be a set of n distinct points $\{p_1, \ldots, p_n\}$ that reside in the interior of Q. Define the *Voronoi Partition* of the convex polygon Q, generated by P to be the set of all points in Q such that all points in the region $V_i(P)$ are closer to p_i than any other point in Q,

$$V_i(P) = \{q \in Q \mid \|q - p_i\| \leq \|q - p_j\|, \forall p_j \in P\}.$$

An example of a Voronoi partitioning is shown in Figure 2.1. Dividing up the area to be explored in this way, keeps us from assigning robots to specific regions which may not be implementable in a decentralized fashion [39]. Using dynamic Voronoi partitions each robot can compute its partition with only knowledge of its neighbors' locations. Thus, using Voronoi partitioning facilitates decentralized control designs. Another advantage to using Voronoi partitions is in the case where there is a robot or sensor failure. Because the Voronoi partitions are made dynamically, the team can adjust their Voronoi partition configuration taking into account their new *neighbors* excluding the failed robot. This procedure insures that all regions in the area will be covered by a corresponding robot.

2.2 Global Optimization

There exists a number of algorithms that can solve the global optimization problem for a general (non-convex) function such as a branch and bound technique, evolutionary algorithms, and randomized algorithm methods [40]. However, because our motivation for this research is focused on a decentralized framework, many of these global optimization techniques are not applicable because of computation power needed or because of the on-line nature of the control algorithm. Also another key point to note is that the probability of detection (POD) map, which models where we should prioritize our search, is updated as the search progresses. As regions are searched by the robots, the POD map is decreased in those regions within the agents sensing radius. This could possibly make the POD map vary greatly from one point that has been searched to an adjacent point that has not been searched by the agent. This limits our ability to make any claims as to certain properties the map (function to be optimized) will exhibit, *e.g.,* Lipschitz continuity or if one can compute lower bounds for feasible regions, which are needed for branch and bound techniques. Also, evolutionary algorithms cannot strictly guarantee convergence to a global optimum. As a result of the limitations imposed by the decentralized, on-line design of our approach, we propose the use of a randomized algorithm [40] to compute an approximate solution to the global optimization problem. A randomized algorithm does not make any assumptions on the function to be optimized or the feasible set of possible solutions and can be efficiently computed. Here we will state the general Monte Carlo method used in our approach [41].

Let the objective function $g(x)$, $x \in A \subset \mathbb{R}^N$ and the set A be measurable where $g^*(x)$ is the global maximum of $g(x)$. To approximate the global maximum $g^*(x)$,

- Generate $X_1, \dots, X_{\mathcal{N}}$ independent identically distributed (i.i.d.) samples from a p.d.f. $f(x)$ such that $f(x) > 0$.

- Find $Y_k = g(X_k),\ k = 1, \dots, \mathcal{N}$.

- Estimate g^* by $\tilde{g}^* = \max(Y_1, \cdots, Y_{\mathcal{N}})$.

It is known that in general there does not exist a stopping condition for an achieved accuracy $\tilde{g}^* - g^* < \epsilon$ for the approximate maximum to the global optimization problem [42]. However one can look for stopping conditions to the global optimization problem in statistical terms.

The probability of sampling at least one point inside the region of attraction of the global maximum when taking $\mathcal{N}$ sample points from A is given by

$$Pr = 1 - (1 - \alpha_1)^{\mathcal{N}}, \tag{2.1}$$

where α_1 is the probability of sampling a point in the region of attraction in one trial. By choosing a lower bound on α_1 and a required “accuracy” level Pr, the number of function evaluations $\mathcal{N}$ needed to achieve a $\tilde{g}^*$ within the accuracy level can be calculated [42].

2.3 Multi-Robot Navigation

Over the years there has been extensive research in robotic motion planning focused on a single agent, [1]. Recently techniques for multi-robot motion planning such as flocking [43], formation control [44], and rendezvous problems [45],[46] have made their way into the literature. One technique to multi-robot motion planning which we focus on in this paper is that of navigation functions. A navigation function is an artificial potential function with a unique minimum located at a goal point whose domain of attraction includes the entire domain excluding the points covered by obstacles. The construction of such a navigation function was first shown by Rimon and Koditschek [47], and slight variations have since made their way into the literature [48], [49]. A recent approach to navigation functions has combined robot navigation and communication constraints [50]. One drawback to these particular constructions of navigation functions is that they are difficult to compute in general and also do not easily lend themselves to general shaped obstacles. To overcome this limitation we consider the construction of a navigation function of the form in [51], which relaxes the requirements on the navigation function, primarily that the gradient of the navigation function need only be piecewise continuous and that the navigation function at the boundaries need not be uniformly maximum. The construction of a navigation function in this way still guarantees a unique minima at the goal point.

Navigation functions involving multiple robots generally have each robot consider all other robots to be moving obstacles [48]. In our approach however, we partition the area among the robot team members, then each robot creates its own navigation function based on its Voronoi partition V_i, the known obstacles in the area C_{obs}, and the goal point calculated from the global optimization algorithm. In this way each robot does not need to consider other robots as obstacles, since by construction the Voronoi partition V_i is unique to robot p_i. Our approach has the advantage of eliminating the need to keep track or "predict" where these "dynamic" obstacles are or will be in the future.

A navigation function, $f(p, g_p, d)$ which is a function of p, the robots current position, the goal point g_p, and points inside the Voronoi partition not occupied by obstacles $d \in D$, can be created by the following procedure [51].

- Make a graph out of the rectangular mesh of the obstacle grid map, with vertices at the corners of each square and edges along the square edges. Remove vertices and edges that are in the interior of obstacles.
- One of the vertices is chosen as the goal point, determined from the global optimization algorithm.
- Solve the shortest-path problem in the graph. Mark each vertex with the corresponding path length, and let this length be the value of $f(\cdot)$ at the vertex.
- Divide the squares into triangles by drawing a diagonal through the corner with the highest $f(\cdot)$ value.
- In the interior of each resulting triangle, let $f(\cdot)$ be a linear interpolation between $f(\cdot)$ at the three vertices.

Note that the shortest-path problem in the graph can be solved with polynomial time algorithms [52]. Also the shortest-path problem to create this navigation function is solved for a 4-neighborhood grid, i.e., L_1 distance.

Alternative planning methods for addressing obstacle avoidance are based on cell decomposition approaches. Cell decomposition is a well-known obstacle avoidance method that decomposes the obstacle-free robot configuration space into a finite collection of non-overlapping convex polygons, known as cells, within which a robot path is easily generated. Although it is computationally intensive, its advantage over other robot path-planning approaches, such as roadmap or potential field methods, is that, under proper assumptions, cell decomposition is resolution complete. Exact cell decomposition is guaranteed to find a free path, whenever one exists, and otherwise to return failure. Its disadvantages are that it is computationally intensive in high-dimensional configuration spaces (e.g., robot manipulators), and that it does not typically allow the user to incorporate other motion constraints, such as, nonholonomic dynamics, or sensing/communication constraints. Also, it is not directly applicable to cooperative networks, in which the path of one robot is influenced by that of the other agents in the network.

2.4 Graph Connectivity

Many definitions given in this section can be found in texts on modern graph theory such as [53]. We begin by considering a heterogeneous team of agents consisting of n sensing agents and m relay agents. For our mathematical formulation we consider each agent x_i to have the following dynamics:

$$\dot{x}_i = Ax_i + Bu_i, \tag{2.2}$$

where A is the system matrix, B is the input matrix, u_i is the input, and $i = 1, \cdots, n+m$. Without loss of generality, let x_i denote the position of agent i. The network of agents described by the system (2.2), gives rise to a *dynamic graph* $\mathcal{G}(x)$.

Definition 2.4.1. *(Dynamic Graph): We Call $\mathcal{G}(x) = (\mathcal{V}, \mathcal{E}(x))$ a dynamic graph consisting of*

- *a set of vertices $\mathcal{V} = \{v_1, \cdots, v_{n+m},\}$ indexed by the set of agents, and*
- *a set of edges $\mathcal{E}(x) = \{(i,j) | d_{ij}(x) < \delta\}$, with $d_{ij} = \|x_i - x_j\|_2$ as the Euclidean distance between agents i and j, and $\delta > 0$.*

Definition 2.4.2. *(Path of a Graph): A* path *is a sequence of distinct vertices such that consecutive vertices share a common edge.*

Definition 2.4.3. *(Graph Connectivity): An non-empty graph $\mathcal{G}$ is called* connected *if any two of its vertices are linked by a path in $\mathcal{G}$.*

Definition 2.4.4. *(Tree): A tree is an undirected graph such that any two vertices are connected by exactly one simple path.*

Definition 2.4.5. *(Spanning Tree): Given a non-empty, undirected, and connected graph $\mathcal{G}$ with vertices $\mathcal{V} = \{v_1, \cdots, v_{n+m},\}$, then a spanning tree of $\mathcal{G}$ is a subgraph which is a tree that connects all vertices, $\mathcal{V} = \{v_1, \cdots, v_{n+m},\}$ together.*

Definition 2.4.6. *(Minimum Spanning Tree): A minimum spanning tree (MST) is a spanning tree that has a weight equal to or less than the weight of any other spanning tree. Note that the minimum spanning tree need not be unique.*

Definition 2.4.7. *(Euclidean Minimum Spanning Tree): A Euclidean minimum spanning tree (EMST) is a minimum spanning tree such that the edge weights between vertices are taken to be the Euclidean distance.*

Definition 2.4.8. *(Adjacency Matrix): Given a non-empty graph $\mathcal{G}$ with vertices $\mathcal{V} = \{v_1, \cdots, v_{n+m},\}$ and edges in the set $\mathcal{E}$, we define the adjacency matrix $A = (a_{ij})$ such that, $a_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$, and $a_{ij} = 0$ otherwise.*

Definition 2.4.9. *(Graph Laplacian): Given a non-empty graph $\mathcal{G}$ with vertices $\mathcal{V} = \{v_1, \cdots, v_{n+m},\}$ and edges in the set $\mathcal{E}$, the graph laplacian is then, $L(x) = D - A$ where D is the valency or degree matrix, $D = diag(\sum_{j=1}^{n+m} a_{ij})$ and A is the adjacency matrix.*

Definition 2.4.10. *(Algebraic Connectivity): Let $\lambda_1 \leq \ldots \leq \lambda_n$ be the ordered eigenvalues of the Laplacian Matrix $L(x)$, then $\lambda_2 > 0$ if and only if $\mathcal{G}(x)$ is connected. $\lambda_2 > 0$ is also known as the algebraic connectivity of the network.*

With these definitions, we can view the dynamic graph induced by the sensor network in a matrix representation. This allows for a straight forward check if the network is connected at any given configuration. Viewing the connectivity of the network from a graph-theoretic point-of-view, the question now becomes how to control the agents (vertices) such that the dynamic graph, $\mathcal{G}(x)$, induced by the agents remains connected throughout the mission.

Chapter 3

Prioritized Sensor Detection

As a first approach, in this chapter we present our multi-vehicle coordination algorithm found in [41] that deals with a connected homogeneous sensor network. The algorithm combines Voronoi partitions, which divide the area to be searched among the robot team members, a Monte Carlo optimization technique, which is used to calculate goal points that correspond to points inside the Voronoi partitions that have the highest probability of containing "good" information, and a modified navigation function that steers the robots from their current positions to their respective goal points while avoiding collisions with the environment and robot team members.

3.1 Control Algorithm for Prioritized Sensor Detection

From design, the control algorithm is decentralized and executed in parallel on all robots. Here we assume n sensing agents which have a fixed and connected communication topology. Before we state the algorithm, a few bits of notation are needed.

Let the probability of detection (POD) map, $M(q)$, reflect the probability of detecting "good information" over the area to be searched. Define β, to be a parameter that reflects the reduction in the probability of detection map for points inside each robots sensing radius, *i.e.,* β is a measure of the reduction of the probability of finding useful information after a region has been visited by a robot. Define the set $D_i = V_i \cap Q_{free}$. D_i represents the set of all points in V_i (Voronoi partition) that are not occupied by an obstacle. The general scheme is outlined below.

1. (*Voronoi region*) For $i = 1, \cdots, n$ determine the Voronoi partition V_i in Q.

2. (*Global optimization*) Apply the general Monte Carlo optimization method over the set D_i to determine an approximate maximum $\tilde{g}^*$ in D_i of $M(q)$.

3. (*Check feasibility*) Determine if $\tilde{g}^*$ is reachable by solving the shortest-path problem from the graph that creates the navigation function. If the point is reachable set the goal point $g_{pi} = \tilde{g_i}^*$. If $\tilde{g_i}^*$ is not reachable then go to Step 2 and determine

   $$\tilde{g_i}^* = \max(Y_1, \ldots, Y_{N-1})$$

   where we exclude $\tilde{g_i}^*$ that was calculated from the previous optimization.

4. (*Navigation function*) Create a navigation function, $f_i(p_i, q_{pi}, d)$ in D_i with g_{pi} set as the goal point.

5. (*Control actuation*) Let $u_i = \dot{p}_i$ and apply the control

   $$u_i = \begin{cases} -k\nabla f_i(\cdot) & \text{if } |u_i| < u_{\max} \\ \\ -\frac{k\nabla f_i(\cdot)}{\|k\nabla f_i(\cdot)\|} \cdot u_{\max} & \text{otherwise} \end{cases} \tag{3.1}$$

   where $k = |p_i - g_{pi}|$ the distance of the robots current position p_i to the goal point g_{pi}.

6. (*Local map update*) For all points that are inside the sensing radius R, update the M as,

$$M(p_i) = \beta M(p_i).$$

7. (*Global map update*) Robot i communicates with its neighbors and exchanges its current position. All robots update their local maps with all other robots local maps to create a synchronized global map.

8. (*Termination*) Check if $t \geq T_{search}$ if true, stop. Else goto Step 1. T_{search} is taken to be the time allowed for the search.

In step 3 we check the feasibility of the goal point obtained from the optimization algorithm. This is done because there may exist configurations of obstacles in the Voronoi partition that cause points in V_i to be unreachable by the robots. If the goal point obtained from the optimization algorithm is unreachable, that point is disregarded and the next best point is taken. It is key to note that there will always exist a feasible goal point for all the robots because in the most improbable case the goal point would be the current robot position. Notice that this case does not however cause robots to become "stuck," since at each iteration of the algorithm the Voronoi regions are updated based on the new positions of the robots at the subsequent optimization is done over the new Voronoi partitions. Also notice that in step 7 we assume that all robots can synchronize their local maps to create a global map. This may seem like somewhat of a limitation on the ability of the control algorithm to be implemented in a decentralized fashion, however it has been shown that this can be done using only one-hop neighbors in the communication network [54]. Although partitioning the search space among the robot team members does place a constraint on the optimal solution, its usefulness is seen in the fact that it keeps multiple robots from visiting adjacent points near the global optimum which may be searched by a single robot as well as insuring collision avoidance among the robots. The partitioning also facilitates a more expansive search in the initial stages of the algorithm.

3.2 Properties of Control Algorithm

Proposition 3.2.1. *(Safety) The control algorithm outlined above guarantees collision avoidance with the environment as well as with other robot team members.*

Proof. Notice that by construction, for any robot configuration $P = \{p_1, \ldots, p_n\}$ the accompanying Voronoi partitions $V_1, \ldots, V_n$ are disjoint. Also by construction the navigation function, $f_i(\cdot)$ is defined over only $D_i = V_i \cap Q_{free}$ for all n robots. Therefore any robot i starting inside D_i and following the control described in (3.1) will stay inside D_i. This guarantees no inter-robot collisions will occur. Obstacle avoidance within D_i comes directly from the construction of the navigation function and the control law described in equation (3.1). □

It is also key to note that convergence to the goal point is also guaranteed using a navigation function and the control law (negated gradient) in equation (3.1), [47]. This is true since by construction the navigation function $f_i(\cdot)$ contains only one minimum at the goal point. A well known result from calculus guarantees that following the negative gradient of a function containing a single minimum will converge to the unique minimum.

3.3 Multi-Sensing Framework

As an extension to [41] we introduce the idea of a multi-sensing framework, where robotic platforms are equipped with more than one sensing modality. This multimodal approach becomes very practical in such applications as search and rescue or hazardous contaminations where more than one factor may play a role in decision making. It may be more beneficial to have a radiation map as well as a temperature map of the area in question before sending in rescue/clean up teams. The problem now becomes, how does one handle prioritizing searching/sensing with multiple sensing capabilities.

From our previous work [41] we have shown how to prioritize searching/sensing efforts through the use of a probability of detection map. To address the multi-sensing framework we propose the use of logistic regression to express the contributions of each sensing modality and its factor on the overall probability of detection (POD) map. This approach allows one to weigh a particular sensing behavior over another depending on need or usefulness of the sensing data.

The logistic function is defined in the following way,

$$f(y) = \frac{1}{1 + e^{-y}}, \tag{3.2}$$

where y is defined as

$$y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_k x_k. \tag{3.3}$$

In equation (3.2), the output represents the probability of a particular outcome. For our purposes $f(y)$ in (3.2) represents the probability of finding useful information at a particular point in the area or interest. In (3.3), the parameters $\alpha_1, \alpha_2, \ldots, \alpha_k$ are the regression coefficients which describe the contribution of the risk factors, $x_1, x_2, \ldots, x_k$. α_0 represents the probability of finding useful information given all risk factors are equal to zero. For our purposes risk factors will be different sensors' probability of detection maps, $M_i(q)$. In this sense, through logistic regression we are able to fuse a variety of different sensing objectives, $M_i(q)$ (POD maps) into a single objective that is weighted based on user need or the mission objective. Figure 3.1 shows a one-dimensional example of combining multiple POD maps. Notice that

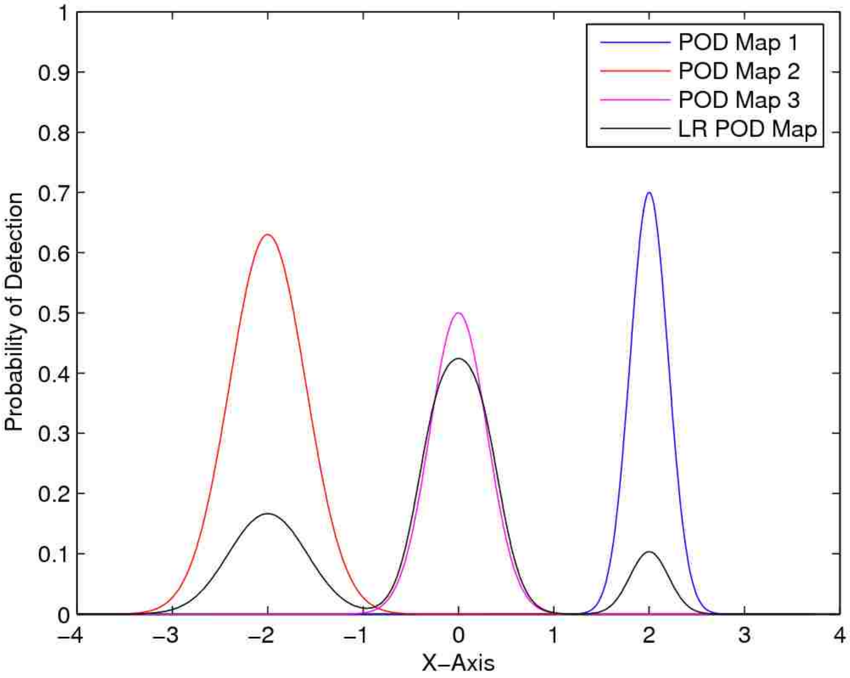

Figure 3.1: One-dimensional example of the linear regression algorithm. Three POD maps (red, magenta, blue) with high probabilities of containing good information at $x = -2, 0,$ and 2 respectively, are combined into a single one-dimensional POD map (black) that reflects the overall mission objective. Here the POD map with the lowest probability (magenta) is the most important to the mission objective which is reflected after the linear regression is done.

although the map represented by the blue line has the highest POD value initially, when combined with the other POD maps it has the lowest POD value. This is due to its regression coefficient, which is taken to be 0.6. In this way, although there is a good indication that some useful information may be there, it may not be a primary objective of the mission. Through this linear regression technique the user can bias what the robots should search/sense first but also allow the possibility of searching/sensing secondary objectives if time permits.

3.4 Task Decomposition Revisited

Through experimentation of the prioritized sensing behavior, [41] we have found that in practice, the Voronoi partitions created by the sensing agents do not always agree throughout the robot network. This is due to localization errors among the agents and also possibly due to corrupted position information being passed among the agents. This seemingly small problem could have very big effects on the outcome of the sensing mission. If the Voronoi partitions overlap, the collision avoidance property of the control algorithm may be violated. If there is separation between the regions, critical areas may never be searched by the agents.

Although an investigation was carried out to determine the best action to overcome the boundary inconsistencies of the Voronoi partitions, it was apparent that this problem was outside the scope of this research. One solution however, may come in the form of more sophisticated localization techniques. In our initial implementation only odometry measurements were used to localize the sensor network. Odometry is known to be inherently noisy and inconsistent due to wheel slippage. A more sophisticated approach utilizing a Kalman filter and more robust sensors such as GPS, laser range finders, an inertial measurement unit, etc. would produce a better estimate of the position and orientation of the agents in the network. A more accurate position and orientation estimation would then lead to more accurate Voronoi boundaries across the network.

Another possible approach may include some type of cooperative localization similar to [55] where a catadioptric camera system is used to determine reliable direction vectors from one agent to all other agents in its field of view. Based on reliable direction and distance vectors the Voronoi boundaries can be computed accurately among the agents in the network. Another approach to cooperative localization may use a single known landmark similar to [56]. Whichever approach is chosen to overcome localization errors, our current framework can incorporate these changes to allow for more accurate Voronoi boundaries.

Chapter 4

Prioritized Search and Adaptive Tracking

Although we have applied the prioritized search algorithm to environmental sensing situations, because of its general formulation it allows itself to other scenarios such as prioritized search for maneuvering targets of interest. This chapter gives a brief description of a book chapter submitted to the American Institute of Aeronautics and Astronautics [57].

We present a framework that combines high-level motion planning and low-level real-time control for a team of UAVs tasked with searching and tracking a target of interest. The high-level motion planning controller can be seen as a hybrid system where the team of UAVs utilize a prioritized search of the area to seek out regions having the highest probability of containing the target-of-interest and switch behaviors to an adaptive sampling-based filter for target tracking when the target enters a UAV's Field Of View (FOV). The low-level control of UAVs is done by utilizing a nonlinear model and controller for quadrotor stabilization and way point tracking. The proposed framework also allows for "human-in-the-loop," which allows a human operator manual control of the UAV team.

The topic of networked UAVs for target search and tracking, although not entirely new, has begun to gain interest in the controls community. Several approaches to reconfiguring the UAV network have been investigated. A technique for dynamically reconfiguring search spaces in order to enable Bayesian autonomous search and tracking missions with moving targets is presented in [58]. The authors in [59] develop a framework for a group of fixed-wing UAVs to cooperatively search and localize static targets using Bayesian filtering techniques and receding-horizon path planning. A somewhat similar work to [59] can be found in [60]. The difference is that [60] takes into account a dynamical model of quadrotors (i.e., UAVs), investigates several sensing modalities, and puts more emphasis on developing scalable distributed estimation techniques. Particle filtering under intermittent information for multi-agent scenarios is investigated in [61]. Aforementioned approaches use probabilistic approaches that enable planning techniques based on minimization of entropy (i.e, uncertainty). Drawbacks of the Bayesian inference can be found, for example, in [62] or in an interesting discussion initiated in [63].

The work in [64] presents a rule-based intelligent guidance strategy for autonomous pursuit of mobile targets by UAVs in an area with threats, obstacles, and restricted regions. A least-square estimation and kinematic relations are used to estimate/predict the target states based on noisy position measurements. However, only generic and simplified models/behaviors of both targets and UAVs are considered. A tightly integrated systems architecture for a decentralized Cooperative Search, Acquisition, and Track (CSAT) mission management algorithm along with actual hardware flight tests is presented in [65]. In a complimentary paper [66], the authors propose an architecture similar to the one we propose in this chapter, however this work does not consider the "human-in-the-loop" scenario and also assume simplified target models.

Our main concern is to design a framework that combines the high-level coordination/path planning of the UAV team and the low-level real-time control and also allows for “human-in-the-loop.” This framework fills a gap in the literature which in general considers the high-level coordination and low-level control independently. Our framework is a bridge between these two approaches which provides a comprehensive top to bottom integration of high-level coordination and low-level control, taking into account sophisticated UAV and target models.

4.1 Problem Formulation

Consider a team of n kinematic agents (UAVs), $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, each equipped with a sensor/camera having a circular Field Of View (FOV) $\mathcal{B}_j$, each tasked with starting from some initial configuration and navigating to visit regions within the area-of-interest (AOI) which contain the highest probability of containing a target-of-interest first, while avoiding collisions with other UAVs. We assume there exists a target-of-interest in the AOI. When the target-of-interest is identified, the UAVs should track the identified target while minimizing their use of resources (sensor use and computational load).

Let the probability of detection (POD) map reflect the probability of detecting a target over the AOI. Define $\beta > 0$ to be a parameter that reflects the reduction in the probability of detection map for points inside each robot's FOV and $\delta > 0$ that can be considered as a forgetting factor of past measurements. Consider the AOI, $Q \subset \mathbb{R}^3$, with boundary ∂Q, to be a simple convex polygon. Let us denote the POD map as $M(q, t)$, where $q \in Q$. Here we assume that Q and its boundary ∂Q is known *a priori* by all UAVs. From the high-level motion coordination perspective, each UAV, p_j is assumed to be holonomic with dynamics:

$$\dot{p_j} = u_j, \ j = 1, \cdots, n \tag{4.1}$$

where $u_j \in \mathbb{R}^3$ is the control input of agent j. From the low-level real-time controller perspective a twelve state dynamic model is used. In this formulation we assume that the UAVs are faster than the targets-of-interest and UAVs fly at a constant altitude above obstacles in the AOI. Also connectivity is assumed between the UAVs and the fusion center.

The target, $\mathcal{T}$, is assumed to be a unicycle unmanned ground vehicle (UGV) that performs purely stochastic maneuvers within the AOI:

$$\dot{x} = \begin{bmatrix} \dot{x}_{\mathcal{T}} \\ \dot{y}_{\mathcal{T}} \\ \dot{\theta}_{\mathcal{T}} \end{bmatrix} = \begin{bmatrix} v_s \cos\theta_s, \\ v_s \sin\theta_s, \\ \omega_s, \end{bmatrix} \tag{4.2}$$

where ω_s and v_s are stochastic input given to the UGV such that $v_s \in [v_{\min}, v_{\max}]$ and $\omega_s \in [\omega_{\min}, \omega_{\max}]$. Modeling the target-of-interest as a unicycle with stochastic inputs allows us to address three main behaviors seen in pursuit evasion games: adversarial, non-adversarial, and collaborative.

4.2 Proposed Approach

Our approach to this particular problem is to develop a framework that combines the high-level motion planning of the group and the low-level control and stabilization of the UAVs. This approach enables the UAVs to bypass the coordination step and only have to communicate with a fusion center to obtain waypoint information. Although this may seem to be a drawback in the sense of a one-point failure, this approach allows for an easy transition from an autonomous mode to a semi-autonomous mode when a "human-in-the-loop" may be needed. This is an important consideration in target search/tracking and surveillance scenarios where human intuition can play a vital role in mission completion. Figure 4.1 shows a diagram of how the framework is structured and implemented. We believe that allowing for a "human-in-the-loop" is an important element to our framework because human intuition or the human perspective is something that can be very useful in search and surveillance scenarios.

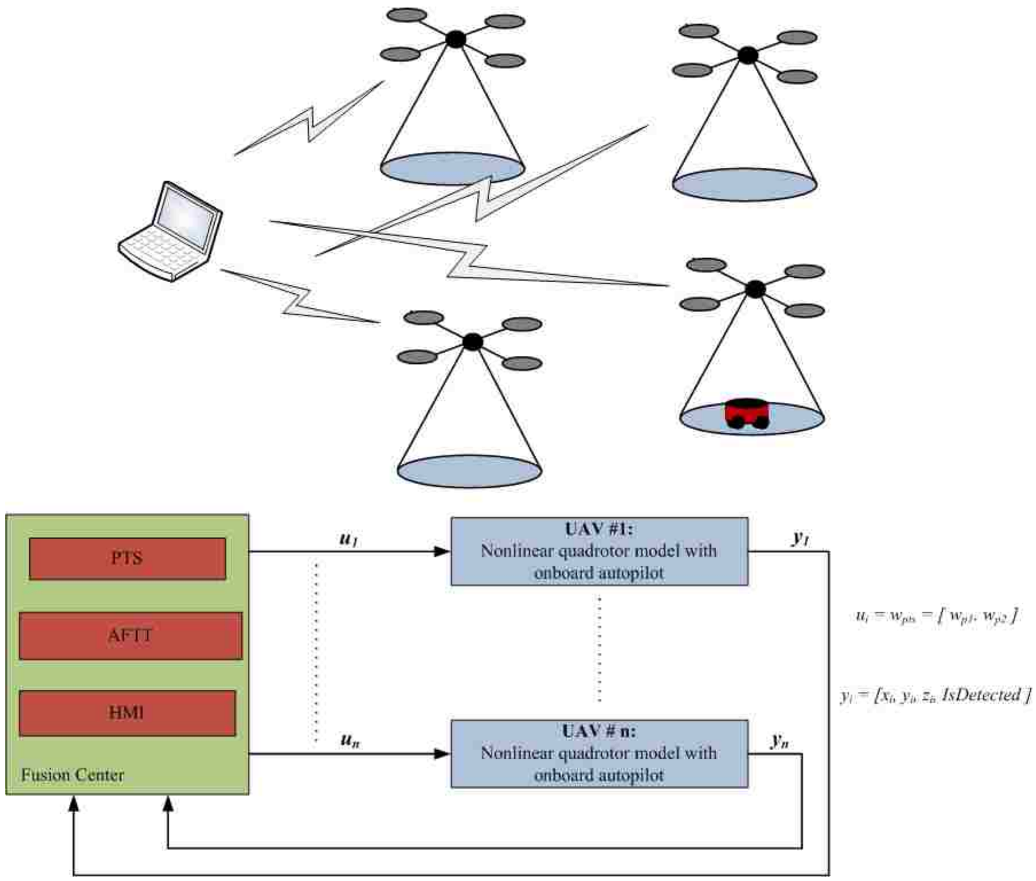
PTS
AFTT
HMI
Fusion Center
u_1
u_n
UAV #1: Nonlinear quadrotor model with onboard autopilot
UAV # n: Nonlinear quadrotor model with onboard autopilot
y_1
y_n
$u_i = w_{pts} = [\, w_{p1}, w_{p2} \,]$
$y_i = [x_t, y_t, z_t, IsDetected\,]$

Figure 4.1: Diagram of the framework the UAVs will employ to acquire and eventually capture a target of interest.

To transition from one behavior to another we introduce two boolean variables that act as flags for the fusion center. Initially the variable $IsDetected =$ **false** and the UAVs employ the prioritized target search (PTS) algorithm. When the target is detected by a UAV, $IsDetected =$ **true** and the fusion center switches controllers to the adaptive sampling-based filter for target tracking (AFTT).

Initially the variable $OverWrite$ = **false**, which is a flag that allows a human operator to control the UAV team manually through inputs of desired waypoints. We denote this by human machine interface (HMI). At any point during the prioritized search or adaptive target tracking, the variable $OverWrite$ can be changed to **true**, to enable human intervention if needed.

The algorithm policy used by the fusion center to determine which algorithm, A, to use to determine inputs for the UAV team is outlined below:

$$A = \begin{cases} \text{PTS} \quad \text{if } \mathit{IsDetected} = \textbf{false} \;\&\; OverWrite = \textbf{false} \\ \text{AFTT} \quad \text{if } \mathit{IsDetected} = \textbf{true} \;\&\; OverWrite = \textbf{false} \\ \text{HMI} \quad \text{if } OverWrite = \textbf{true}. \end{cases}$$

4.3 Simulations

To simulate our heterogeneous framework we assume a rectangular AOI (30m×30m) with four UAVs having FOV's of 1.5m, 2.0m, 2.1m, and 2.5m. Maximum target linear and angular velocities are set to 0.7m/s and 1.5rad/s respectively with UAV maximum velocity set at 5m/s. The probability of detection map parameters β and δ are set to 0.4 and 5×10^{-3}. In Figure 4.2 we see that three areas have high probabilities of containing the target-of-interest initially. The UAVs are shown in magenta as well as their FOVs. The PTS algorithm is implemented to identify points in the AOI that have the highest probability of containing the target-of-interest. Waypoint information is relayed to the onboard low-level controller to navigate the UAVs to those regions for searching.

When the target-of-interest is detected by one of the UAVs, their behavior changes from the PTS algorithm to the AFTT algorithm for target tracking. Figure 4.3 shows the reduction of the probability of detection map when the UAVs switch from the PTS algorithm to the AFTT algorithm as well as their respective trajectories while

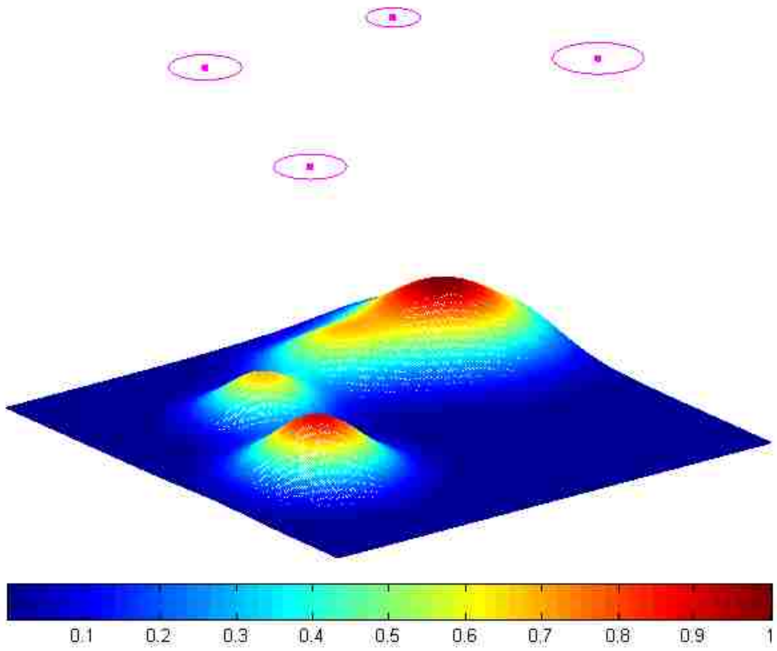

Figure 4.2: The initial probability of detection map. UAVs and their respective FOVs are shown in Magenta.

implementing the PTS algorithm.

Figure 4.4 shows a snapshot of the AFTT algorithm. The target-of-interest is shown as a blue arrow and its trajectory is the blue hashed line. The red dotted line shows the fitting circle used by the AFTT algorithm. Notice that the fitting circle is a line, indicating a circle with almost infinite radius. The magenta hashed line shows the tracking UAV's trajectory. During this simulation only one UAV was actively tracking, however the AFTT algorithm allows for "handing off" the target to other UAVs if the target enters their FOV.

Comparison of waypoint tracking between a holonomic UAV model and the dynamic quadrotor model is shown in Figure 4.5. The trajectories of the holonomic and dynamic quadrotor model are very similar, indicating the low-level controller

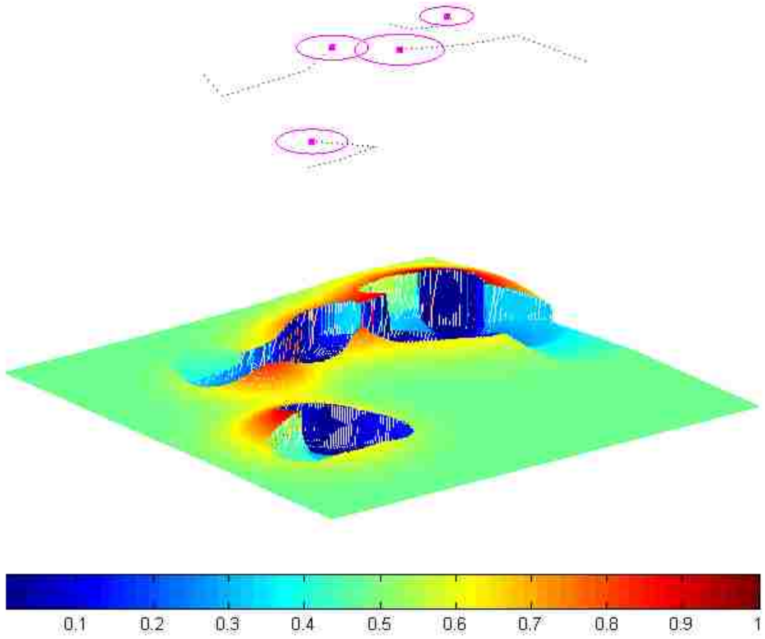

Figure 4.3: The reduced probability of detection map at the moment the UAVs switch from the searching algorithm (PTS) to the target tracking algorithm (AFTT). UAVs are shown in Magenta and their trajectories are shown by dotted black lines.

does a good job of steering the quadrotors to the desired waypoints. Figures 4.6 and 4.7 show the dependence of quadrotor velocity to the distance from the desired way point. The velocity is high when a new waypoint is received, and then decreases as the quadrotor approaches the desired waypoint. In future implementations we will design a path following algorithm which will enable speed profile requirements.

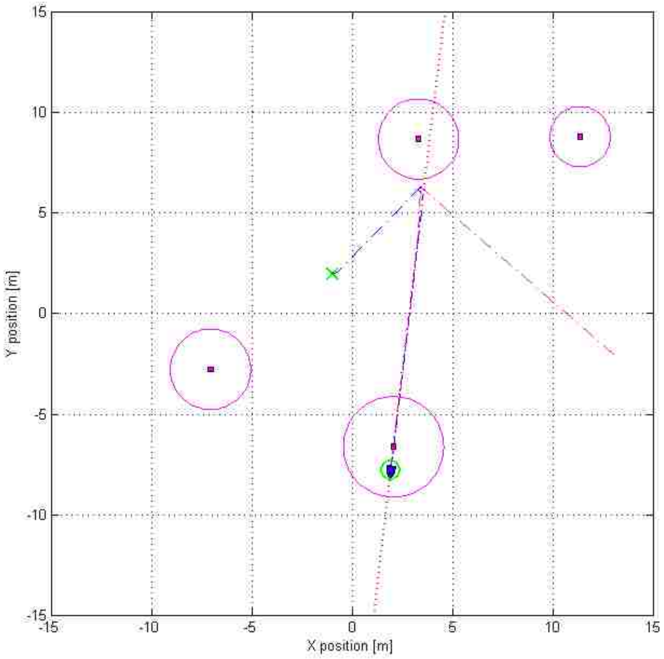

Figure 4.4: Snapshot of the tracking algorithm. The target is seen as a blue arrow. The fitting circle from AFTT algorithm is shown as a dotted red line. The blue line is the target trajectory and the magenta line is the trajectory of the UAV that is currently tracking.

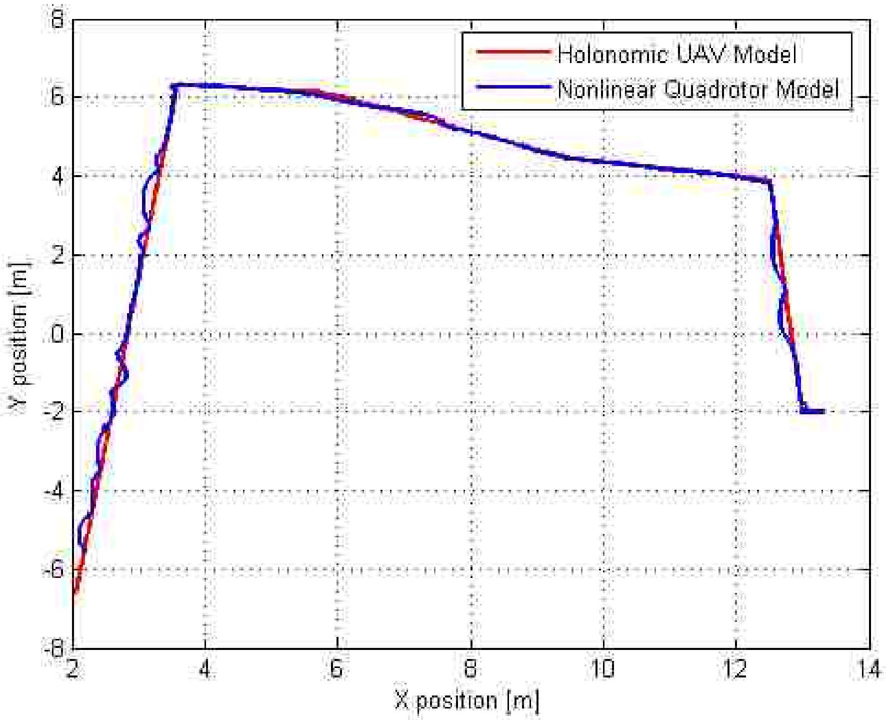

Figure 4.5: Comparison of waypoint tracking for holonomic UAV model and nonlinear quadrotor model.

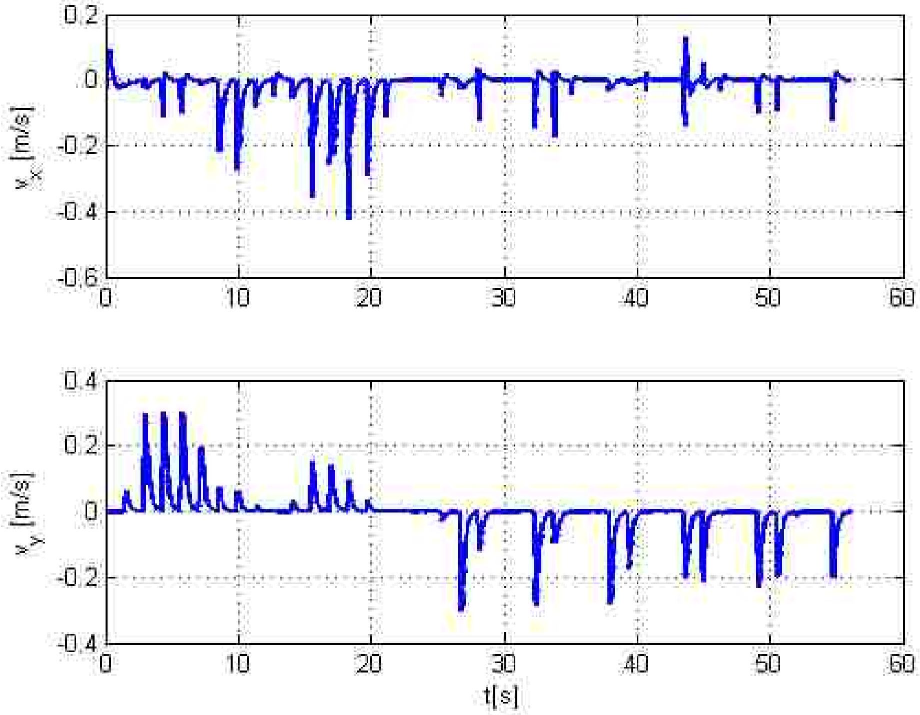

Figure 4.6: Velocities of nonlinear quadrotor model (x and y components) during waypoint tracking.

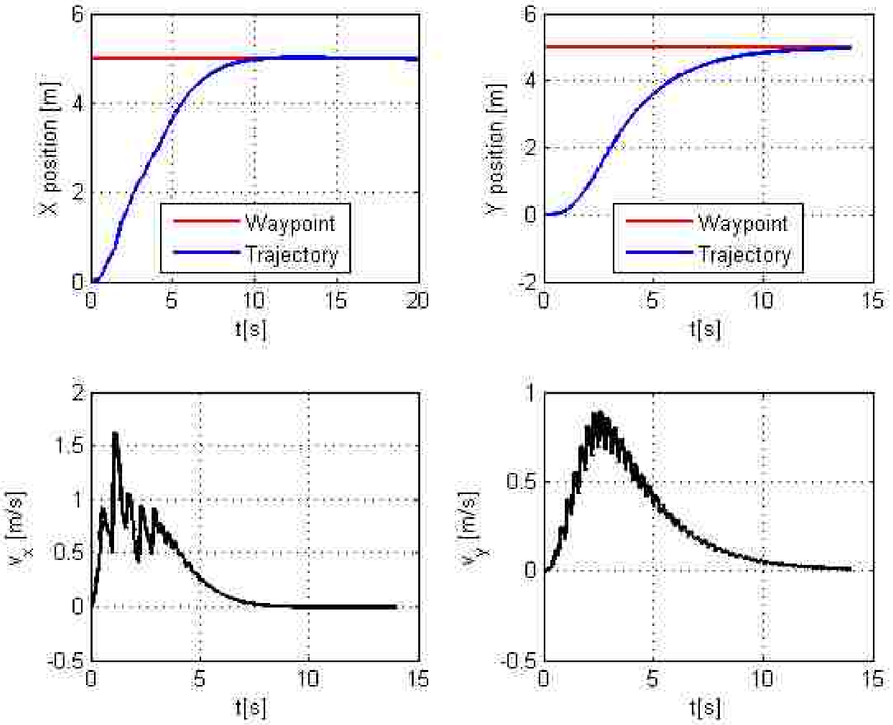

Figure 4.7: Quadrotor velocity profile for a single waypoint. Both x and y components are shown.

Part II: Heterogeneous Sensor Network

Chapter 5

Connectivity Maintenance of a Heterogeneous Sensor Network

In this chapter we discuss our approach to multi-robot connectivity maintenance. We propose adding specialized agents that are better equipped (hardware) to relay information over longer distances within the sensor network. This allows the sensor network to have a longer "reach" in the search space. It also allows the sensing agents to be built in a way that the communication hardware can be minimized. This may have benefits in hardware costs and also the size and shape of the sensing agents in the network.

5.1 Heterogeneity in Reconfigurable Sensor Networks

In this section we try to provide a discussion of what distinctions should be made in the literature to better represent the idea of a heterogeneous system. This distinction will help categorize heterogeneity in such a way that will allow for a better characterization of the problem being approached.

Heterogeneity, in general, can be used to describe a system that is comprised of members that have variations. In reconfigurable sensor networks these variations may be as obvious as platform variations, *i.e.,* a network of UAVs and UGVs, or as subtle as the actual sensor footprint of the sensing nodes. The title of a "heterogeneous system" is adequate to describe both types of systems. Although this is a fair description for both systems, the term heterogeneity doesn't describe how these variations come together to help solve an overall mission objective.

We propose to categorize heterogeneity into two main categories. The first being hardware-based heterogeneity and the second one is considered objective-based heterogeneity. Hardware heterogeneity is just that, variations in the hardware between members of the sensor network. This includes differing sensor footprints, communication ranges, vehicle models (platforms), sensing modalities, among many others. Hardware-based heterogeneity has its difficulties in deriving controllers that can handle these hardware variations while achieving an objective such as sensor coverage [67].

On the other end of the spectrum is objective-based heterogeneity where multiple sub-objectives must be met to achieve the overall mission objective. A simple example may be a team of agents that need to search an area while also making sure any possible exits from the area are covered by a sensing agent. The team may split up, with a few patrolling possible exits while others do the search. We see that the difficulties in objective-based heterogeneity is combing behaviors in a way that there is cohesiveness as well as maintaining properties such as convergence, stability, etc.

Figure 5.1 gives a representation of how we view heterogeneous systems. In this brief discussion, it is seen that heterogeneity by itself is not descriptive enough to

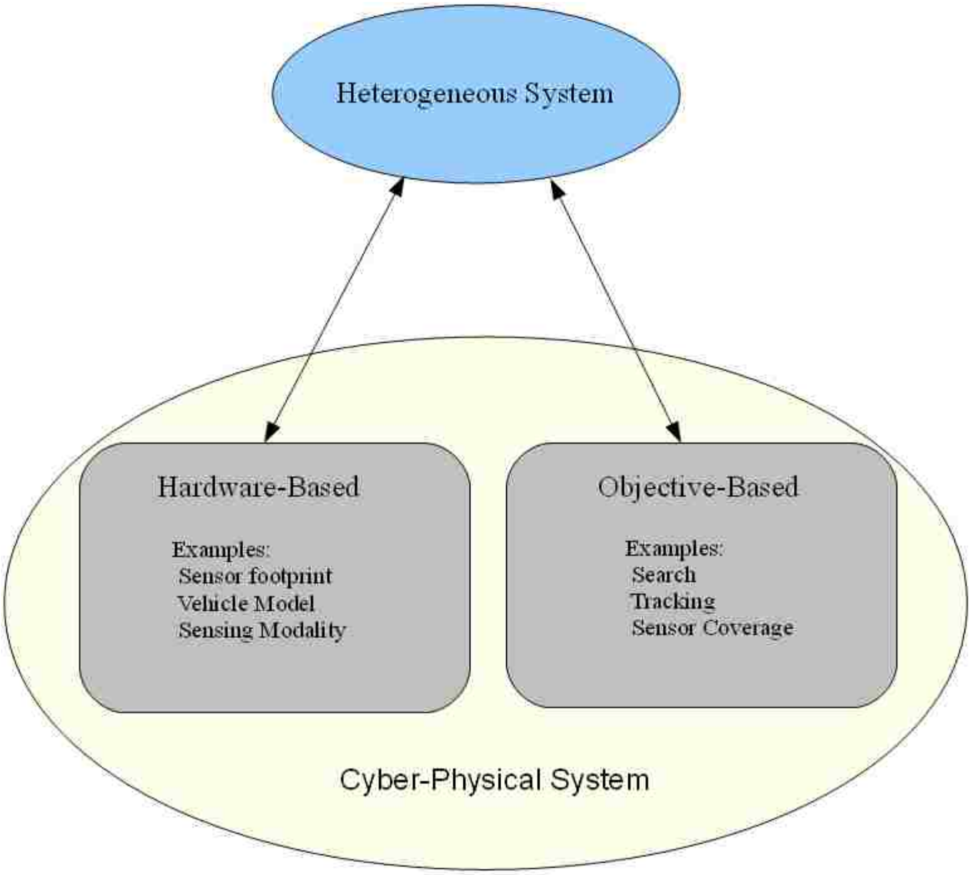

Figure 5.1: Characterization of heterogeneous systems within reconfigurable sensor networks.

explain the variations within system. We believe our categorized representation is a more detailed way of characterizing the variations within a heterogeneous system that will lead to a better understanding of the difficulties involved as well as the issues being addressed for a particular heterogeneous system.

In the particular problem we are considering in this dissertation, the heterogeneity of our sensor network falls under both hardware-based and objective-based heterogeneity. Our problem exhibits hardware-based heterogeneity since we are considering two types of vehicles, UAVs and UGVs, as well as agents with different communication ranges, *i.e.,* relay and sensing agents. We also consider our system to have objective-based heterogeneity from the fact that the sensing agents objective is purely sensing the environment, while the relay agents objective is to react to the sensing agents and keep the network connected.

From the fact that our approach involves a cyber-physical system, it is not surprising that our system exhibits both hardware-based and objective-based heterogeneity. This can be seen from the tight integration of various capabilities for its operation and interaction with the physical environment. There exists a need in the robotics community to develop frameworks for cyber-physical systems, which undoubtedly will involve addressing difficulties that arise within heterogeneous systems. With any cyber-physical system, heterogeneity will be a key issue that must be addressed.

5.2 Problem Formulation

We begin by considering a heterogeneous team of agents consisting of n sensing agents, which we will consider for this application as Unmanned Ground Vehicles (UGVs), and m relay agents, which we will assume are Unmanned Aerial Vehicles (UAVs), in two and three dimensions. Assume the n sensing agents are equipped with sensors capable of sensing an environmental phenomena within a finite radius R_s and communicating within a finite radius $R_c(q) \leq R_{c_{max}}$. Here we assume that the communication radius will change based on the positions of the robots. This relaxation in the communication range allows us to model, to some degree, the path loss in the communication channel [50]. Incorporating communication channel characteristics, which has been largely ignored in the literature to date, allows for a better system model. Also let us assume that the m relay agents are capable of communicating over a finite radius R_{rc} such that $R_{rc} > R_{c_{max}}$ *i.e.*, the relay robots are better equipped for communication than the sensing agents and the relay robots communication range is not dependent on location. Consider the area of interest Q, assumed to be a simple convex polygon with boundary ∂Q, including its interior. Define C_{obs} as the union of all obstacles in the region Q, and let $Q_f = Q \setminus C_{obs}$ be the area within Q that is free of obstacles. Let us define the probability of detection map $M(q)$, which reflects the probability of detecting an environmental phenomena over the area to be searched. For our mathematical formulation we consider each agent x_i to have the following dynamics:

$$\dot{x}_i = Ax_i + Bu_i, \tag{5.1}$$

where A is the system matrix, B is the input matrix, u_i is the input, and $i = 1, \cdots, n+m$. We are assuming a linear controllable system under the premise that the dynamics from both ground vehicles as well as aerial vehicles with an autopilot system can be conservatively estimated in such a way. This is an abstraction to the real dynamics of both ground and aerial vehicles with the assumption that there exists well tuned low-level controllers for each vehicle type. Here we are considering our sensor network to be heterogeneous not only because relay and sensing agents have different communication ranges but also because they play different roles in the sensor network.

5.3 Communication Constraints

In our scenario there exists three particular communication link possibilities. The first being, *relay/sensor* communication, where a sensor communicates directly to a relay agent. The second, *relay/relay* communication, where a relay shares a communication link with another relay agent. The last communication link possibility is *sensor/sensor* communication where sensors communicate directly with each other. For the following formulation let us consider the case where the communication radius of the sensing robots is not location dependent, *i.e.* $R_c(q) = R_c$. Note that each agents communication range describes the range over which the agent can both send and receive information. For ease of notation let us also consider that the relay agents fly at constant altitude, h, and their communication range will be taken as its two dimensional projection at this constant height.

Similar to the work on homogeneous networks of Bullo *et al.*, [68], we now formulate the *connectivity constraint set* for each particular communication link possibility of our heterogeneous network based on the geometry of the communication radii. For the following definitions we will use $\bar{\mathcal{B}}(p, r)$ to denote a closed ball of radius r centered at p in $\mathbb{R}^2$.

Definition 5.3.1. *(*Relay/Sensor *connectivity constraint set) Consider two agents, one relay agent i located at position p_i and one sensing agent j located at position p_j such that $||p_i - p_j||_2 \leq R_{rc}$. Then the connectivity constraint set of agent i with respect to agent j is*

$$\Upsilon_{d_{rs}}(p_i, p_j) = \bar{\mathcal{B}}(\frac{p_i + p_j}{2}, \frac{R_{rc}}{2}). \tag{5.2}$$

Figure 5.2 shows an example of the relay/senosr connectivity constraint set.

Definition 5.3.2. *(*Relay/Relay *connectivity constraint set) Consider two relay agents, one agent i located at position p_i and one agent j located at position p_j such that $\|p_i - p_j\|_2 \leq R_{rc}$. Then the connectivity constraint set of agent i with respect to agent j is*

$$\Upsilon_{d_{rr}}(p_i, p_j) = \bar{\mathcal{B}}(\frac{p_i + p_j}{2}, \frac{R_{rc}}{2}). \tag{5.3}$$

Figure 5.3 shows an example of the relay/relay connectivity constraint set.

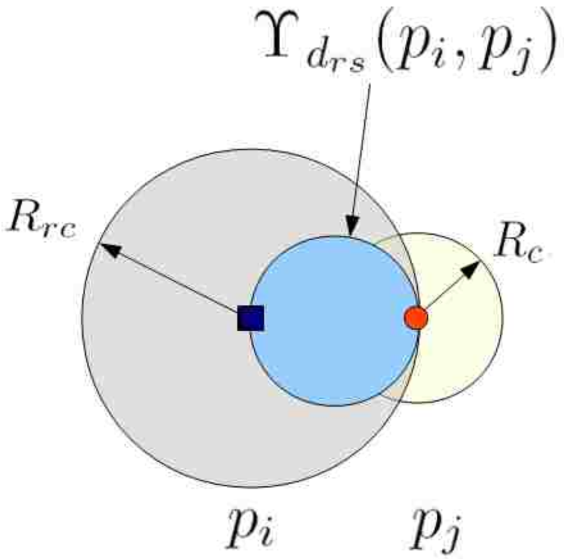

Figure 5.2: Motion constraints set for sensor/relay connection. The red dots represents the relay and sensing agents. The blue disk represents the motion constraint set that guarantees connectivity.

Definition 5.3.3. *(*Sensor/Sensor *connectivity constraint set) Consider two sensing agents, one agent i located at position p_i and one agent j located at position p_j such that $\|p_i - p_j\|_2 \leq R_c$. Then the connectivity constraint set of agent i with respect to agent j is*

$$\Upsilon_{d_{ss}}(p_i, p_j) = \bar{\mathcal{B}}(\frac{p_i + p_j}{2}, \frac{R_c}{2}). \tag{5.4}$$

Figure 5.4 shows an example of the sensor/senosr connectivity constraint set.

Remark 5.3.4. *Notice that definition 5.3.1 and definition 5.3.2 are the same. This is due to the fact that sensors communication radius can be ignored when a communication link exists between a sensor and relay, since the sensors communication radius*

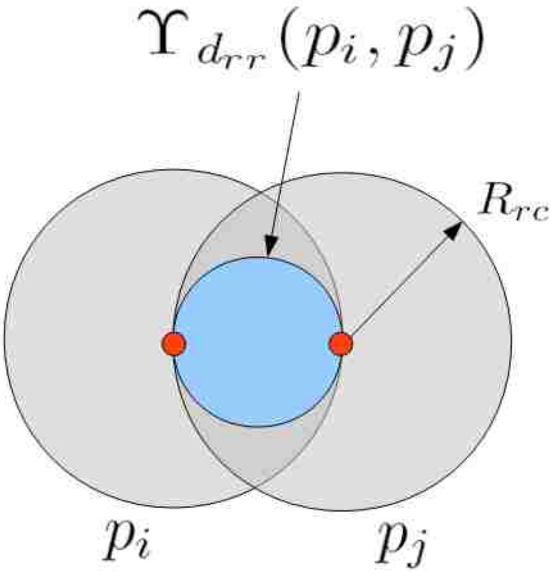

Figure 5.3: Motion constraints set for relay/relay connection. The red dots represents two relay agents. The blue disk represents the motion constraint set that guarantees connectivity.

is smaller that that of the relay agent. This is also a consequence of the assumption that the sending and receiving channels are symmetric.

Definition 5.3.5. *(Connectivity constraint set for relay agent w.r.t. heterogeneous network) Consider a group of agents containing both sensing and relay agents located at $\mathcal{P} = \{p_1, p_2, \ldots, p_{n+m}\}$. Then the connectivity constraint set of relay agent i with respect to all other agents in the group is*

$$\Upsilon_{d_{hr}}(p_i, \mathcal{P}) = \{x \in \Upsilon_{d_{rr}}(p_i, p_j) | q \in \mathcal{P} \backslash \{p_i\} \ s.t. \ \|q - p_i\|_2 \leq R_{rc}\}. \tag{5.5}$$

Figure 5.5 shows an example of a relay connectivity constraint set w.r.t. the heterogeneous network.

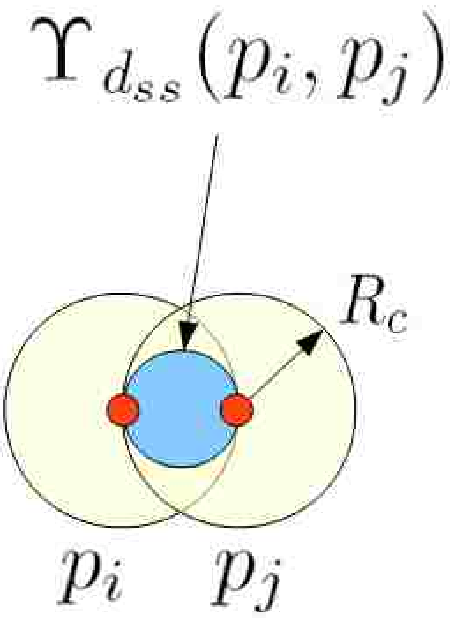

Figure 5.4: Motion constraints set for sensor/sensor connection. The red dots represents two sensing agents. The blue disk represents the motion constraint set that guarantees connectivity.

Before we can state the definition of the connectivity constraint set for a sensing agent with respect to the heterogeneous network we need some preliminaries. Let p_i be a sensing agent, then

$$\Lambda_{ss} = \cap_{j=1}^{n} \Upsilon_{ss}(p_i, p_j), \text{ where } p_j \in \text{ sensors}, \tag{5.6}$$

$$\Lambda_{sr} = \cap_{k=1}^{m} \Upsilon_{sr}(p_i, p_k), \text{ where } p_k \in \text{ relays}. \tag{5.7}$$

Now we can define the connectivity constraint set for a sensing agent with respect to the heterogeneous network.

Definition 5.3.6. *(Connectivity constraint set for sensor agent w.r.t. heterogeneous network) Consider a group of agents containing both sensing and relay agents located at $\mathcal{P} = \{p_1, p_2, \ldots, p_{n+m}\}$. Then the connectivity constraint set of a sensor agent i*

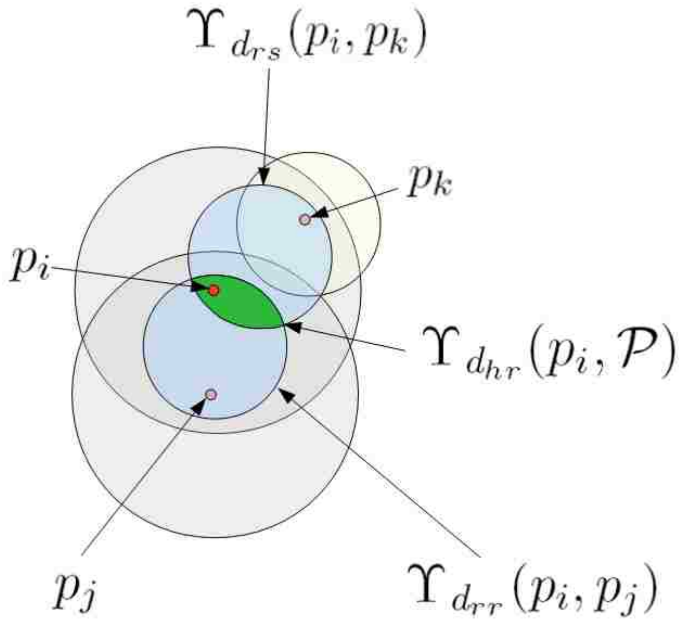
$\Upsilon_{d_{rs}}(p_i, p_k)$
p_k
p_i
$\Upsilon_{d_{hr}}(p_i, \mathcal{P})$
p_j
$\Upsilon_{d_{rr}}(p_i, p_j)$

Figure 5.5: Motion constraint set for a relay agent w.r.t. the network, $\Upsilon_{\mathrm{d}_{hr}}(p_i, \mathcal{P})$. The green area represents the motion constraint set that guarantees connectivity for relay agent p_i w.r.t. the heterogeneous network.

with respect to all other agents in the group is

$$\Upsilon_{d_{hs}}(p_i, \mathcal{P}) = \Lambda_{ss} \cap \Lambda_{sr}. \tag{5.8}$$

Figure 5.6 shows an example of a sensors connectivity constraint set w.r.t. the heterogeneous network. The connectivity constraint sets defined in (5.3.1) - (5.3.6) define the set of allowable positions that each robot may take such that the communication network will remain connected. Thus the connectivity constraint sets defines the feasible motion for each individual robot to remain connected with the network.

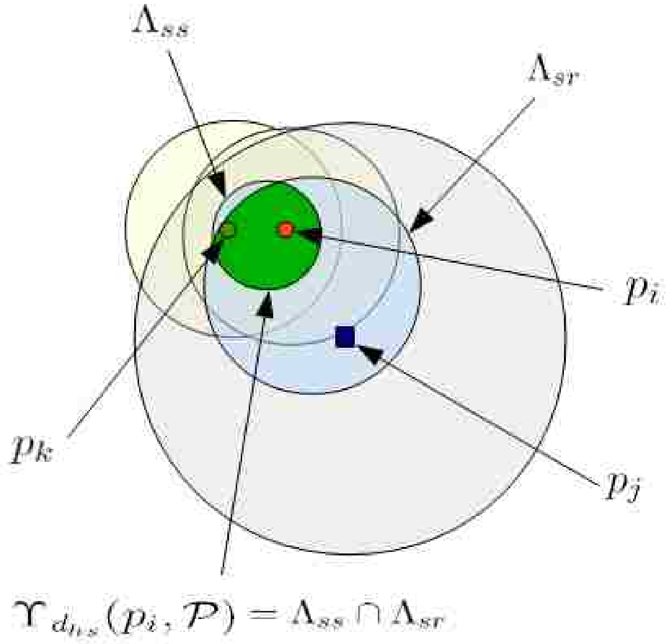
Λ_{ss}
Λ_{sr}
p_i
p_k
p_j
$\Upsilon_{d_{hs}}(p_i, \mathcal{P}) = \Lambda_{ss} \cap \Lambda_{sr}$

Figure 5.6: Motion constraint set for a sensor agent w.r.t. the network, $\Upsilon_{\mathrm{d}_{hs}}(p_i, \mathcal{P})$. The green area represents the motion constraint set that guarantees connectivity for sensor agent p_i w.r.t. the heterogeneous network.

5.4 Heterogeneous Proximity Graph

Due to the heterogeneity of our sensor network, we must define an appropriate proximity graph. As a reminder, a proximity graph describes connections between a set of vertexes based on their relative distances.

Definition 5.4.1. *(Proximity Graph, [68]) Let $S \subset \mathbb{R}^N$. A proximity graph $\mathcal{G}$ associates to a set of distinct points $\mathcal{P} = \{p_1, \ldots, p_n\} \subset S$, an undirected graph with vertex set $\mathcal{P}$ and whose edge set is given by $\mathcal{E}_{\mathcal{G}}(\mathcal{P}) \subseteq \{(p,q) \in \mathcal{P} \times \mathcal{P} | p \neq q\}$.*

We see that due to the heterogeneity of our network, the edge set of our proximity graph should depend on the agent type. The following definition describes how the edge set should be created for our heterogeneous proximity graph.

Definition 5.4.2. *(Heterogeneous r(p)-disk graph, $\mathcal{G}_{disk(r(p))}(\mathcal{P})$) Two agents p_i and p_j are neighbors if they are located within a distance $r(p) = R_c$ if both p_i and p_j are sensing agents or $r(p) = R_{rc}$ if one of the agents is a relay agent, i.e.,*

$$(p_i, p_j) \in \mathcal{E}_{\mathcal{G}_{disk(r(p))}}(\mathcal{P}) \;\; if \; \begin{cases} \|p_i - p_j\|_2 \leq R_c & and\ p_i,\ p_j\ both\ sensing\ agents \\ \|p_i - p_j\|_2 \leq R_{rc} & and\ p_i\ or\ p_j\ is\ a\ relay\ agent. \end{cases} \tag{5.9}$$

An example of the $\mathcal{G}_{\text{disk}(r(p))}(\mathcal{P})$ graph is shown in Figure 5.7. In the $\mathcal{G}_{\text{disk}(r(p))}(\mathcal{P})$ graph, edges depend on the agent distances as well as agent connection combinations.

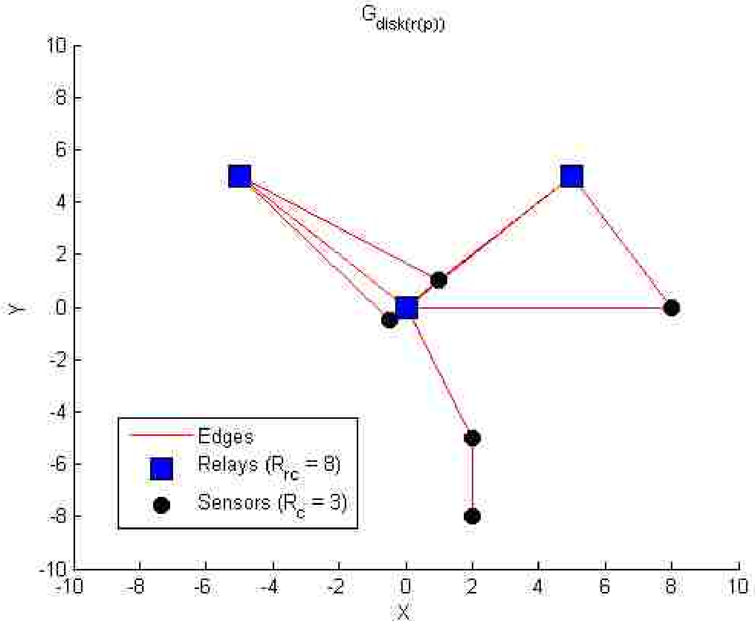

Figure 5.7: Example of Heterogeneous r(p)-disk graph, $\mathcal{G}_{\text{disk}(r(p))}(\mathcal{P})$ with three relay robots (blue squares) and five sensing robots (black circles). The red lines represent edges in the graph between respective agents.

The heterogeneous r(p)-disk proximity graph, $\mathcal{G}_{\text{disk}(r(p))}(\mathcal{P})$, allows us to represent the network topology of our heterogeneous system. It is seen that depending on the configuration of the network there may exist heavy redundancy in the connections (Figure 5.7). This redundancy comes at the cost of more constraints on each agent, therefore reducing the size of the set of possible inputs that guarantee connectivity. This reduction stems from the fact of the intersection of multiple sets.

Let us now define the the weighted complete graph which we will denote as, $\mathcal{G}$ throughout the rest of this paper.

Definition 5.4.3. *(Weighted Complete Graph, $\mathcal{G}$) Let $S \subset \mathbb{R}^N$. The weighted complete graph $\mathcal{G}$ associates to a set of distinct points $\mathcal{P} = \{p_1, \ldots, p_{n+m}\} \subset S$, an undirected graph with vertex set $\mathcal{P}$ and whose edges $e = (p_i, p_j) \in \mathcal{E}_{\mathcal{G}}(\mathcal{P})$ has the following weights $w(e)$,*

$$w(e) = \begin{cases} ||p_i - p_j||_2 + R_{rc} & \textit{if } p_i,\ p_j \textit{ both sensing agents} \\ ||p_i - p_j||_2 & \textit{if } p_i \textit{ or } p_j \textit{ is a relay agent.} \end{cases} \tag{5.10}$$

5.5 Minimizing Motion Constraints

With a formal way of representing the motion constraints for each agent with respect to the heterogeneous group, we now are left with trying to minimize the constraints (links) in such a way that we expand the input set the agents can choose from that still guarantees connectivity at the next time step. One solution is to take $\mathcal{G}_{\mathrm{disk}(r(p))}(\mathcal{P})$ and run a minimum spanning tree algorithm to determine a subgraph of the r(p)-disk graph that has the minimum number of connections needed to remain connected. A key result from modern graph theory is that assuming $\mathcal{G}_{\mathrm{disk}(r(p))}(\mathcal{P})$ is connected their always exist a minimal spanning tree [69]. The usefulness of the minimum spanning tree approach is that it allows us to weigh connections between agents. This may be useful in enforcing *relay/sensor* connections over *sensor/sensor* connections since *relay/sensor* connections offer a greater motion set for the agents as opposed to the *sensor/sensor* connections because of the larger communication radius. Another reason to bias certain network connections when possible is because relay nodes are better equipped to handle communication data, *i.e.,* higher bandwidth. Figure 5.8 shows the Euclidean Minimum Spanning Tree (EMST) for the r(p)-disk graph weighted by the Euclidean distance between connected vertexes, $\mathcal{G}_{EMST,\mathcal{G}}$. It is key to note that the EMST is a subgraph of the $\mathcal{G}_{\text{disk}(r(p))}(\mathcal{P})$ graph and contains the minimal number of connection to maintain a connected graph.

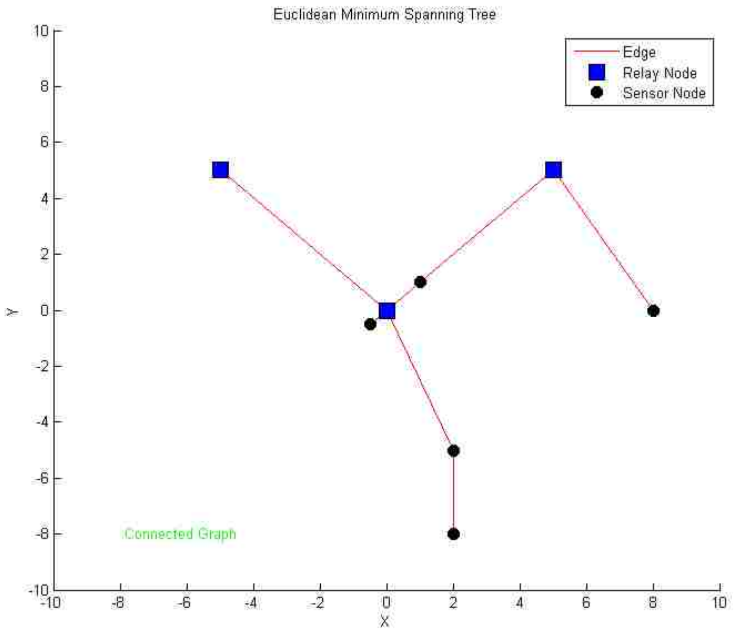

Figure 5.8: Example of $\mathcal{G}_{EMST,\mathcal{G}}$ with three relay robots (blue squares) and five sensing robots (black circles). The red lines represent edges in the graph between respective agents. Notice the graph is connected.

5.5.1 Shaping the Network Configuration

To help bias *relay/sensor* connections over *sensor/sensor* connections with respect to the Minimum Spanning Tree (MST) we now formulate a weighting factor for *sensor/sensor* connections. From definitions (5.3.1) and (5.3.3) we see that the motion constraint set for *relay/sensor* connections is larger than *sensor/sensor* connections due to a larger communication radius. With the help of Figure 5.9 we look at the scenario of one relay and two sensing agents. In terms of the MST, all connections that have a possibility of being biased can be broken down in this way. For ease of notation we will refer to the $MST_{\mathcal{G}_{\text{disk}(r(p))}}$ as just the MST.

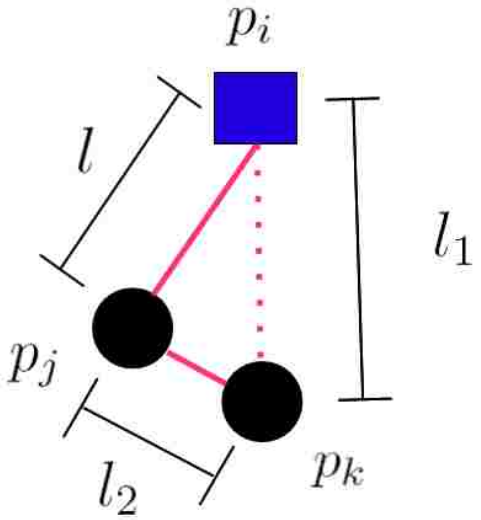

Figure 5.9: Figure of one relay agent (blue square) and two sensing agents (black circle) used to formulate weighting factor for *sensor/sensor* connections.

Let $\|p_i - p_j\|_2 = l$, $\|p_i - p_k\|_2 = l_1$ and $\|p_j - p_k\|_2 = l_2$. Let us assume that $l < l_1 \leq R_{rc}$ and $l_2 \leq R_c$. From construction of the MST, the red solid edges between p_i, p_j, and p_k in Figure 5.9 will be chosen since

$$l + l_2 < l_1 + l_2,$$

$$l + l_2 < l + l_1.$$

Let $\epsilon > 0$ denote the minimum distance between two sensing agents, *i.e.,* physical footprint. Under other circumstance ϵ can also be considered the threshold distance where two sensing agents should communicate directly. To bias the *relay/sensor* connection (red dotted line) a weighting factor ξ_1, must be constructed such that when $l_2 = \epsilon$, $\xi_1 l_2 \geq R_{rc}$. Defining $\xi_1 = \left(\frac{R_{rc}}{\epsilon} + \delta_1\right)$ with $\delta_1 \geq 0$ we get the following,

$$\begin{aligned} \xi_1 l_2 &= \left(\frac{R_{rc}}{\epsilon} + \delta_1\right) l_2 \\ \xi_1 l_2 &= R_{rc} + \delta_1 \epsilon \\ \xi_1 l_2 &\geq R_{rc}. \end{aligned} \tag{5.11}$$

Therefore, with the connection weighting factor ξ_1 we now have the following,

$$l + l_1 \leq l + \xi_1 l_2,$$

$$l + l_1 \leq l_1 + \xi_1 l_2.$$

Weighting the *sensor/sensor* connection (edge) by a factor of ξ_1 allows us to bias the MST to chose the *relay/sensor* connections. Figure 5.10 shows a connected $\mathcal{G}_{\text{disk}(r(p))}$ graph with many redundant connections. Figure 5.11 and Figure 5.12 show the difference in network connections between the MST and the connection weighted MST (MST$_{CW}$) respectively, where *sensor/sensor* connections are weighted by the factor ξ_1. Notice that in the MST$_{CW}$ graph, the *relay/sensor* connections are chosen over *sensor/sensor* connections.

To understand the effect of choosing *relay/sensor* connections over *sensor/sensor* connections, we calculate the area covered by the motion sets for the various graph representations. Figure 5.13 shows the difference in the area of the motion constraint sets for the $\mathcal{G}_{\text{disk}(r(p))}$, $\mathcal{G}_{MST}$, and the $\mathcal{G}_{MST_{CW}}$ graph for each agent in the network. Notice that $\mathcal{G}_{MST_{CW}}$ graph allows for the largest motion constraint set area for sensing agents (vertex 6-13).

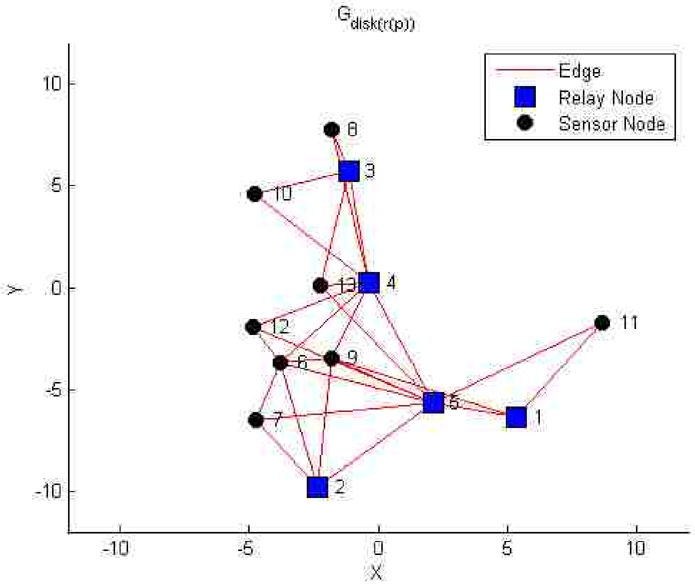

Figure 5.10: The $\mathcal{G}_{\text{disk}(r(p))}$ with many redundant connections.

We sum the total areas of the feasible motion sets for each graph representation of the network in Figure 5.11 and Figure 5.12, and see that the $\mathcal{G}_{\text{disk}(r(p))}$ totalled 167 units^2, the MST totalled 337 units^2, and the MST$_{CW}$ totalled 462 units^2. This gives us a good indication that the MST$_{CW}$ graph “frees” up more area for the sensing agents to investigate than the other network graph representations. This is attributed to the fact that relay agents have a larger communication radius.

In a similar fashion we can bias *relay/relay* connections. This may be advantageous for certain mission objectives or when large amounts of data may need to be transferred directly to a relay node. It may not be efficient or even possible to send large amounts of data through a sensing node to reach another relay node.

Using Figure 5.15, as previously stated let us assume the minimum distance between any two agents is $\epsilon > 0$. Let us also assume that from Figure 5.15 that $l, l_1, l_2 < R_{rc}$ and for convenience assume $l_1 < l_2 < l$. From the point of view of the

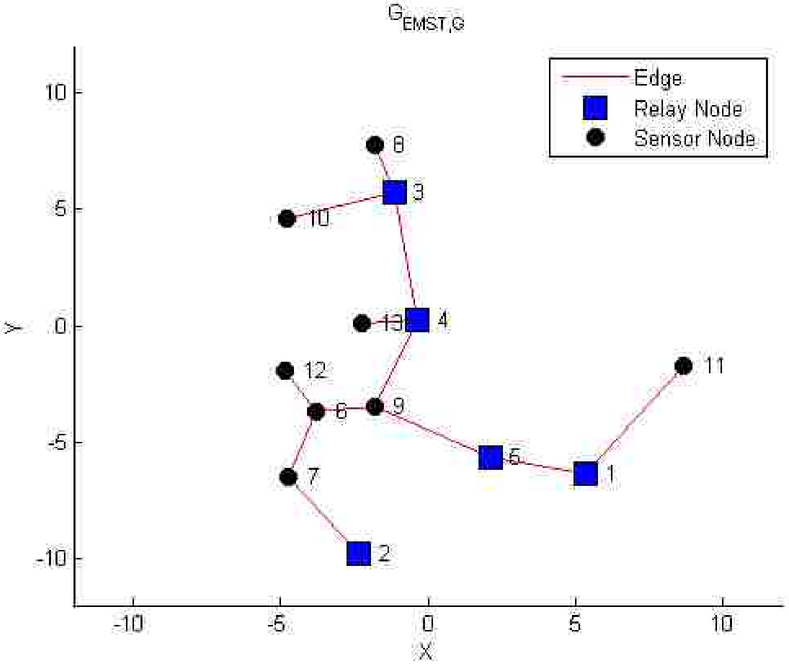

Figure 5.11: Example of the MST for thirteen agents with no connection weights.

MST the red edges between p_i, p_j, and p_k in Figure 5.15 will be chosen since

$$l_1 + l_2 < l_1 + l,$$

$$l_1 + l_2 < l + l_2.$$

To bias direct *relay/relay* connections (Figure 5.15 red dotted line), we use a weighting factor $\xi_2 = \frac{\epsilon}{l}$. Choosing ξ_2 in this way insures that a direct *relay/relay* connection will be chosen over the multi-hop connection by the MST algorithm in Figure 5.15, *i.e.,* relay→ sensor→ relay. This is seen from the fact that,

$$\xi_2 l = \frac{\epsilon}{l} l,$$

$$\xi_2 l = \epsilon.$$

Therefore, now the distance between p_i and p_k is ϵ from the point of view of the MST algorithm. Since the minimum distance of any two agents is ϵ the MST will choose

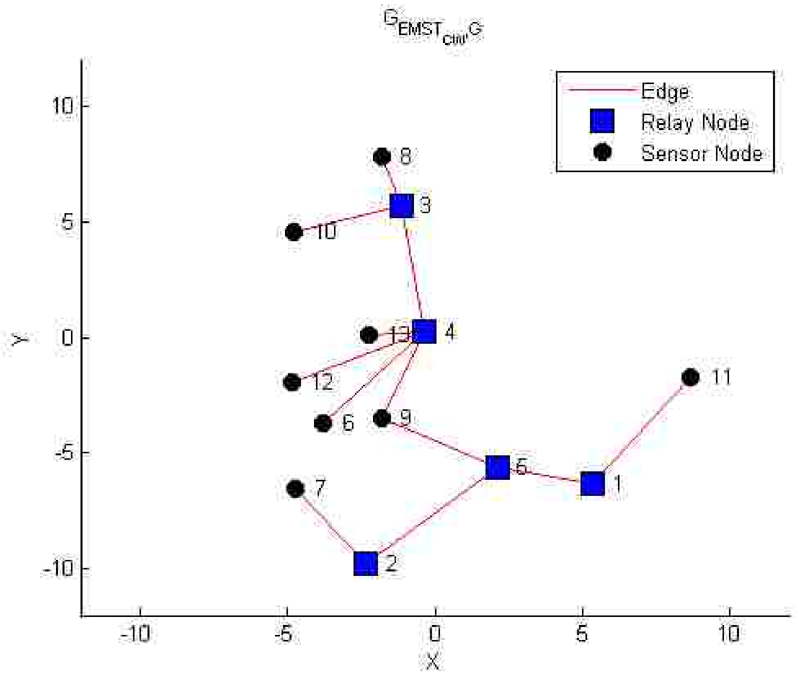

Figure 5.12: The MST$_{CW}$ graph for the thirteen agents. Notice how the *relay/sensor* connections are chosen over *sensor/sensor* connections.

the direct *relay/relay* link. Figure 5.16 shows the network configuration using both ξ_1 and ξ_2 as connection weights (MST$_{CW}$).

Distributed Minimum Spanning Tree

In a classic paper by Gallager *et al.* [70], it was shown that there exists a distributed algorithm to compute the minimum weight spanning tree of a connected, undirected graph with $\mathcal{N}$ nodes and $\mathcal{E}$ edges. It was also shown that at most $5\mathcal{N}\log_2\mathcal{N} + 2\mathcal{E}$ messages need to be passed to determine the minimum weight spanning tree. Much work has focused on improving the time complexity of this algorithm and an approximate MST can be calculated in almost optimal time [71].

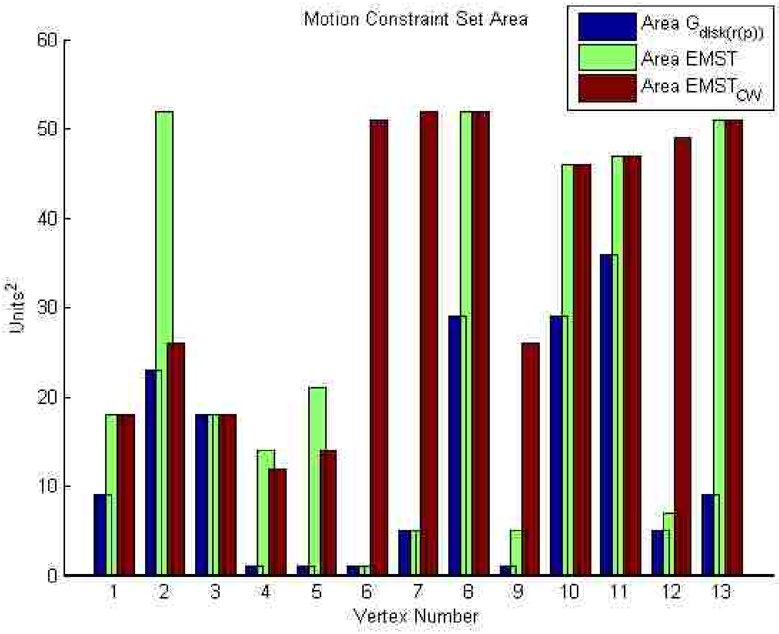

Figure 5.13: Comparison of the area of the motion constraint sets for the $\mathcal{G}_{\text{disk}(r(p))}$, MST, and MST$_{CW}$. Notice that in the MST$_{CW}$ graph the motion constrain set area is higher for sensors (vertexes 6-13) than in the MST graph.

These key results allow the robots to compute the MST of the heterogeneous proximity graph with only local information from adjacent robots. This is very useful because it also allows the feasible motion constraint sets to be calculated in a distributed fashion. For a summary of these distributed algorithms the reader is referred to [70] and [71].

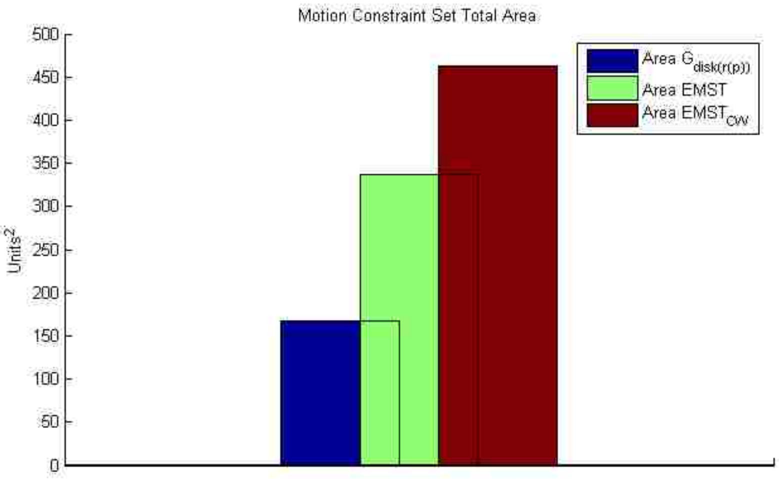

Figure 5.14: Figure showing the total area covered by the motion constraint sets by the three different graph representations.

5.6 Properties of the Heterogeneous Motion Constraints

This section details some properties of the heterogeneous motion constraint for agents described in (5.1).

Theorem 5.6.1. *Consider a relay agent, p_i, with dynamics described in* (5.1)*, such that* (5.1) *is at least stabilizable and having a motion constraint set as defined in* (5.3.1)*. If p_i takes a goal point $g_{p_i} \in \Upsilon_{d_{rs}}(p_i, p_j)$ at time t_1, then p_i will be connected to p_j when it reaches g_{p_i} at time t_2.*

Proof. Given the fact that the dynamics of p_i are at least stabilizable implies that there exists a static control law $u(t) = -Kx(t)$ such that the closed loop system is asymptotically stable, *i.e.*, $\lim_{t\to\infty} x(t) = g_{p_i}$.

By definition $g_{p_i} \in \Upsilon_{\mathrm{d}_{rs}}(p_i, p_j)$ which implies that $\|g_{p_i} - p_j\|_2 < R_{rc}$, hence p_i at

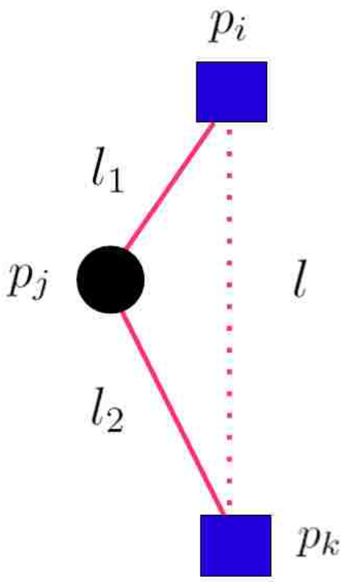

Figure 5.15: Figure of two relay agents (blue squares) and one sensing agent (black circle) used to formulate weighting factor for *relay/relay* connections.

position g_{p_i} at time time t_2 is connected with p_j. □

Theorem 5.6.2. *Consider a relay agent, p_i, with dynamics described in* (5.1)*, such that* (5.1) *is at least stabilizable and having a motion constraint set as defined in* (5.3.2)*. If p_i takes a goal point $g_{p_i} \in \Upsilon_{d_{rr}}(p_i, p_j)$, at time t_1, then p_i will be connected to p_j when it reaches g_{p_i} at time t_2.*

Proof. The proof is similar to the proof in Theorem 5.6.1. □

Theorem 5.6.3. *Consider a sensing agent, p_i, with dynamics described in* (5.1)*, such that* (5.1) *is at least stabilizable and having a motion constraint set as defined in* (5.3.3)*. If p_i takes a goal point $g_{p_i} \in \Upsilon_{d_{ss}}(p_i, p_j)$, at time t_1 then p_i will be connected with p_j when it reaches g_{p_i} at time t_2.*

Proof. The proof is similar to the proof in Theorem 5.6.1. □

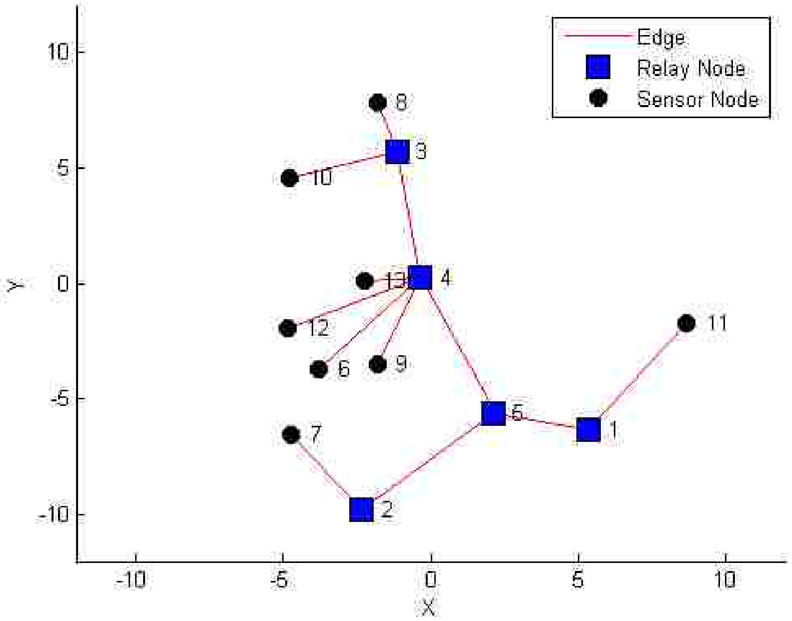

Figure 5.16: Example MST_{CW} for thirteen agents with both *sensor/sensor* and *relay/relay* connection weights. Notice the *relay/sensor* connections are chosen over *sensor/sensor* connections and *relay/relay* connections are chosen.

Theorem 5.6.4. *Consider a relay agent, p_i, with dynamics described in* (5.1)*, such that* (5.1) *is at least stabilizable and having a motion constraint set as defined in* (5.3.5)*. If p_i takes a goal point $g_{p_i} \in \Upsilon_{d_{hr}}(p_i, \mathcal{P})$ at time t_1, then p_i will be connected with all agents at time t_2 that it was connected with at t_1 when it reaches g_{p_i}.*

Proof. Given the fact that the dynamics of p_i are at least stabilizable implies that there exists a static control law $u(t) = -Kx(t)$ such that the closed loop system is asymptotically stable, *i.e.*, $\lim_{t\to\infty} x(t) = g_{p_i}$.

By definition $g_{p_i} \in \Upsilon_{\mathrm{d}_{hr}}(p_i, \mathcal{P})$ which implies that

$$\forall q \in \mathcal{P}\backslash\{p_i\} s.t. \|q - g_{p_i}\|_2 < R_{rc},$$

hence p_i at position g_{p_i} at time time t_2 is connected with all $p_j \in \mathcal{P}$ that it was connected to at time t_1. □

Theorem 5.6.5. *Consider a sensing agent, p_i, with dynamics described in* (5.1)*, such that* (5.1) *is at least stabilizable and having a motion constraint set as defined in* (5.3.6)*. If p_i takes a goal point $g_{p_i} \in \Upsilon_{d_{hs}}(p_i, \mathcal{P})$ at time t_1, then p_i will be connected with all agents at time t_2 that it was connected with at t_1 when it reaches g_{p_i}.*

Proof. Given the fact that the dynamics of p_i are at least stabilizable implies that there exists a static control law $u(t) = -Kx(t)$ such that the closed loop system is asymptotically stable, *i.e.* $\lim_{t \to \infty} x(t) = g_{p_i}$.

By definition $g_{p_i} \in \Upsilon_{\mathrm{d}_{hs}}(p_i, \mathcal{P})$ which implies that

$$\forall q \in \mathcal{P} \backslash \{p_i\} s.t. \left\|q - g_{p_i}\right\|_2 < R_{rc},$$

if q is a relay agent and

$$\forall q \in \mathcal{P} \backslash \{p_i\} s.t. \left\|q - g_{p_i}\right\|_2 < R_{c},$$

if q is a sensing agent. Hence p_i at position g_{p_i} at time time t_2 is connected with all $p_j \in \mathcal{P}$ that it was connected to at time t_1. □

Theorem 5.6.6. *For $R_{rc}, R_c \in \mathbb{R}^+$ and $R_c < R_{rc}$, then $\mathcal{G}_{MST,\mathcal{G}} \subset \mathcal{G}_{disk(r(p))}$ if and only if $\mathcal{G}_{disk(r(p))}$ is connected, where $\mathcal{G}$ is the weighted complete graph described in definition 5.4.3 with vertex set $\mathcal{P} = \{p_1, p_2, \ldots, p_{n+m}\}$.*

Proof. ($\Rightarrow$) If $\mathcal{G}_{\mathrm{MST},\mathcal{G}} \subseteq \mathcal{G}_{\mathrm{disk}(r(p))}$ then $\mathcal{G}_{\mathrm{disk}(r(p))}$ is connected by definition of $\mathcal{G}_{\mathrm{MST},\mathcal{G}}$, *i.e.,* MST is connected.

($\Leftarrow$) (By Contradiction) Assuming $\mathcal{G}_{\mathrm{disk}(r(p))}$ is connected but $\mathcal{G}_{\mathrm{MST},\mathcal{G}} \not\subseteq \mathcal{G}_{\mathrm{disk}(r(p))}$ implies two possible scenarios, (i) there exists two vertices $p_i, p_j \in \mathcal{E}_{\mathcal{G}_{\mathrm{MST},\mathcal{G}}}$ such that

$||p_i - p_j||_2 > R_c$ where p_i, p_j are both sensing agents or,(ii) there exists two vertices $p_i, p_j \in \mathcal{E}_{\mathcal{G}_{\mathrm{MST},\mathcal{G}}}$ such that $||p_i - p_j||_2 > R_{rc}$ where p_i or p_j is a relay agent.

(i) If we remove the edge linking p_i and p_j from $\mathcal{E}_{\mathcal{G}_{\mathrm{MST},\mathcal{G}}}$ then the tree becomes disconnected with two connected components, T_1 and T_2 such that $p_i \in T_1$ and $p_j \in T_2$. Since by assumption $\mathcal{G}_{\mathrm{disk}(r(p))}$ is connected, there must exist $p_k, p_l \in \mathcal{P}$ such that $p_k \in T_1$, $p_l \in T_2$ and $||p_k - p_l||_2 < R_c$ if both p_k and p_l are sensing agents or $||p_k - p_l||_2 < R_{rc}$ if either p_k or p_l is a relay agent. If we add the edge (p_k, p_l) to the set of edges of $T_1 \cup T_2$, then the resulting graph $\mathcal{G}^*$ is acyclic, connected, and contains all vertices in $\mathcal{P}$. This implies $\mathcal{G}^*$ is a spanning tree. Since $||p_k - p_l||_2 \leq R_c < ||p_i - p_j||_2$ we can conclude that $\mathcal{G}^*$ has a smaller length than $\mathcal{E}_{\mathcal{G}_{\mathrm{MST},\mathcal{G}}}$ from the definition of the edge weights of the complete weighted graph $\mathcal{G}$ in definition 5.4.3. This is a contradiction of the definition of the MST.

(ii) Under the same argument as (i) and replacing R_c with R_rc it can be shown similarly that the result is a contradiction of the MST. □

For the $\mathcal{G}_{\mathrm{disk}(r(p))}$ graph we take the edge weights between two connected vertices to be Euclidean distance between the two agents represented by the nodes in the graph. Therefore the MST for the $\mathcal{G}_{\mathrm{disk}(r(p))}$ graph becomes the EMST.

Theorem 5.6.7. *For $R_{rc}, R_c \in \mathbb{R}^+$ and $R_c < R_{rc}$, if $\mathcal{G}_{disk(r(p))}$ is connected then,*

$$\sum_{e \in \mathcal{G}_{MST,\mathcal{G}_{disk(r(p))}}} w(e) \leq \sum_{e \in \mathcal{G}_{MST,\mathcal{G}}} w(e).$$

Proof. From theorem 5.6.6 we have that $\mathcal{G}_{\mathrm{MST},\mathcal{G}} \subset \mathcal{G}_{\mathrm{disk}(r(p))}$ which implies that $\mathcal{E}_{\mathcal{G}_{\mathrm{MST},\mathcal{G}}} \in \mathcal{E}_{\mathcal{G}_{\mathrm{disk}(r(p))}}$. We also know that by definition $\mathcal{E}_{\mathcal{G}_{\mathrm{MST},\mathcal{G}_{\mathrm{disk}(r(p))}}} \in \mathcal{E}_{\mathcal{G}_{\mathrm{disk}(r(p))}}$. Looking at the edge weights of $\mathcal{G}_{\mathrm{MST},\mathcal{G}_{\mathrm{disk}(r(p))}}$ (Euclidean distance) we have,

$$w(e) = \begin{cases} ||p_i - p_j||_2 \leq R_c & \text{if } p_i,\, p_j \text{ both sensing agents} \\ ||p_i - p_j||_2 \leq R_{rc} & \text{if } p_i \text{ or } p_j \text{ is a relay agent.} \end{cases}$$

For the edge weights of $\mathcal{G}_{\text{MST},\mathcal{G}}$ described in definition 5.4.3 we see that,

$$w(e) = \begin{cases} ||p_i - p_j||_2 + R_{rc} & \text{if } p_i,\, p_j \text{ both sensing agents} \\ ||p_i - p_j||_2 & \text{if } p_i \text{ or } p_j \text{ is a relay agent.} \end{cases}$$

Since $R_{rc} > 0$ by definition, the edge weights of all possible links chosen by the MST algorithm for the $\mathcal{G}_{\text{MST},\mathcal{G}}$ graph will be greater than or equal to those chosen for the $\mathcal{G}_{\text{MST},\mathcal{G}_{\text{disk}(r(p))}}$ graph. □

Remark 5.6.8. *In a centralized scenario or when the $\mathcal{G}_{disk(r(p))}$ graph is the complete graph, theorem 5.6.7 provides a straightforward method for checking whether the MST of the $\mathcal{G}_{disk(r(p))}$ was computed correctly.*

5.7 Case Study: Centroidal Heterogeneous Motion Constraint Set Configurations

This section looks at the particular situation when the heterogeneous team moves towards their respective centroid of their feasible motion set. This centroidal configuration is a straightforward way to test the claims of Theorem 5.6.4 and Theorem 5.6.5. It is key to note that the centroidal configuration is just one of many different possible ways of testing the claims of Theorem 5.6.4 and Theorem 5.6.5.

5.7.1 Simulations

By construction, the centroid $\mathcal{C}_i^*$ of each agents motion constraint set (MCS_i) lives in the interior of its motion constraint set. Therefore by setting $\mathcal{C}_i^*$ as the goal point for each agent $i, \forall i = 1, \ldots, n+m$, then the heterogeneous team should remain connected when each goal point is reached by the respective agents. Figure 5.17 depicts the centroidal heterogeneous motion constrain set configuration for two relay agents and one sensing agent. The red stars denote the centroid of each agents constraint set.

Algorithm 1 Centroidal Behavior ($g_{p_i} = \mathcal{C}_i^*$)

```
while $t < t_{\text{final}}$ do
  for $x_i = 1, \ldots, n+m$ do
    Calculate $\mathcal{C}_i^*$ from $MCS_i$ (Equations (5.2)-(5.8))
    $g_{p_i} \Leftarrow \mathcal{C}_i^*$
    while $\Delta t < T_s$ do
      $u_i(t) = -Kx_i(t)$
    end while
  end for
end while
```

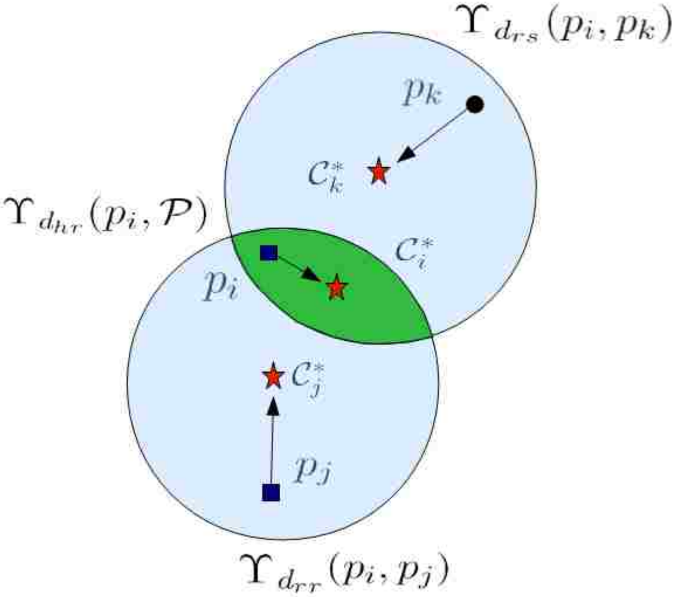
$\Upsilon_{d_{rs}}(p_i, p_k)$
p_k
$\mathcal{C}_k^*$
$\Upsilon_{d_{hr}}(p_i, \mathcal{P})$
$\mathcal{C}_i^*$
p_i
$\mathcal{C}_j^*$
p_j
$\Upsilon_{d_{rr}}(p_i, p_j)$

Figure 5.17: Figure showing the centroidal heterogeneous motion constraint set configurations. Each respective agent calculates its own centroid w.r.t. its constraint set and then moves towards it.

For this simulation we set $R_c = 3$m, and $R_{rc} = 10$m. We use $m = 4$ relay agents and $n = 10$ sensing agents initially in a random configuration but in such a way that the heterogeneous team is initially connected. Each agent $i, \forall i = 1, \ldots, n + m$, then uses its neighbors of the $\mathcal{G}_{MST_{CW}}$ graph to calculate the centroid of their respective motion constraint set. Each agent is modeled as a double integrator (5.1), and a state feedback control law is used to drive the agents from their current position to their respective goal points $(\mathcal{C}_i^*)$. Every $T_s = 0.05$ seconds, position information is exchanged among the team members and an updated $\mathcal{G}_{MST_{CW}}$ graph is calculated. Based on the new information, new centroids are updated and used as the goal point. An outline is given in Algorithm 1. The simulation lasts for a total of five seconds.

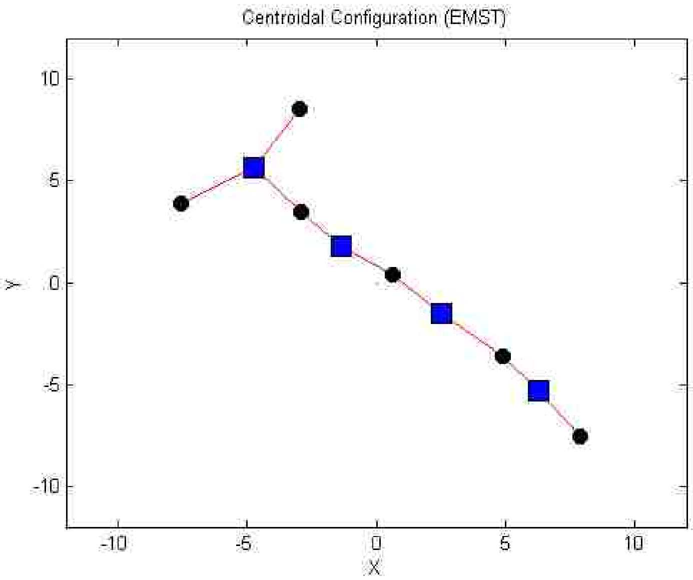

Figure 5.18: Centroidal configuration for a heterogeneous team that moves towards goal points that are the centroid of their respective motion constraint set.

Figure 5.18 show the final configuration of the heterogeneous team after five seconds of the centroid seeking behavior. Figure 5.19 and Figure 5.20 show the connectivity with respect to the second smallest eigenvalue criteria $(\lambda_2(\mathcal{G}) > 0 \Rightarrow$

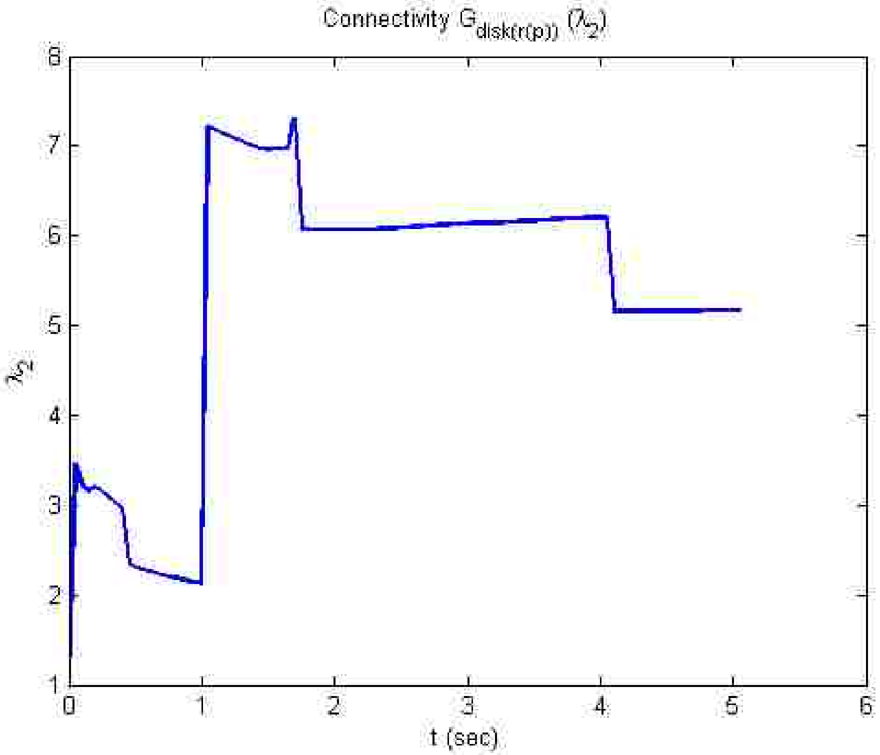

Figure 5.19: Graph of the second smallest eigenvalue for the $\mathcal{G}_{\text{disk}(r(p))}$ graph for the centroidal behavior.

$\mathcal{G}$ is connected) for the $\mathcal{G}_{\text{disk}(r(p))}$ and $\mathcal{G}_{MST_{CW}}$ graph respectively. We see that both graphs stay connected at all times, however only the $\mathcal{G}_{MST_{CW}}$ graph is used to calculate the constraint sets. Note that the larger $\lambda_2\left(\mathcal{G}_{\text{disk}(r(p))}\right)$, the more connections exist in the graph. Figure 5.19 shows that in the $\mathcal{G}_{\text{disk}(r(p))}$ graph there exist redundant links that allows for some robustness to node failures, however it is unclear at this point to what extent this is true.

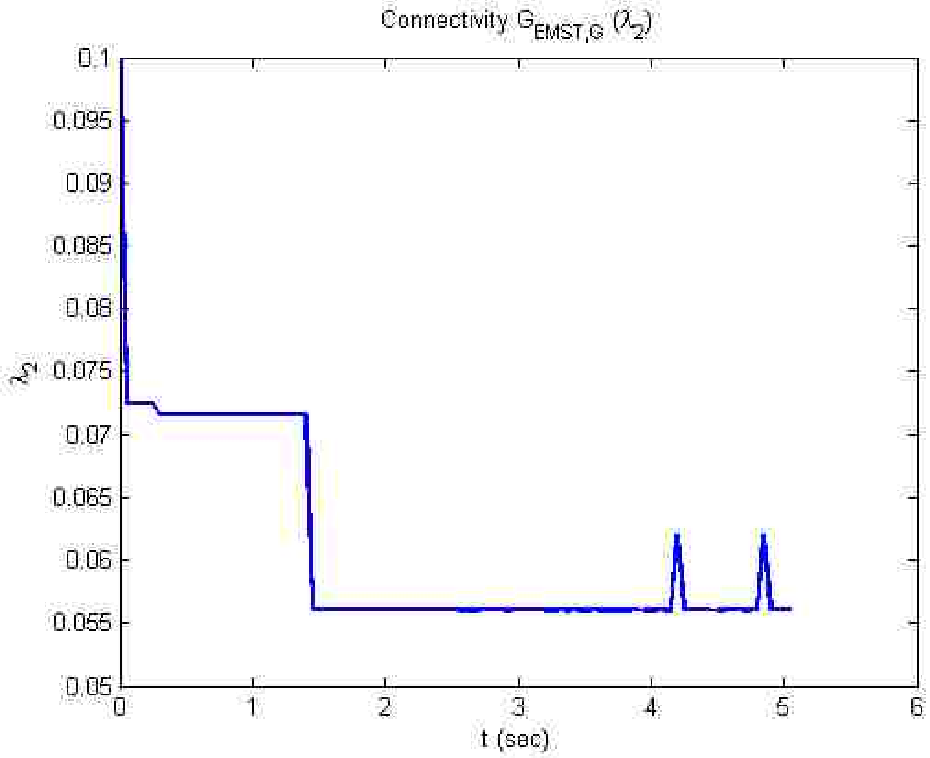

Figure 5.20: Graph of the second smallest eigenvalue for the $\mathcal{G}_{MST,\mathcal{G}}$ graph for a simulation of the centroidal behavior.

5.7.2 Node Redundancy and Network Robustness

We can see from Figure 5.18 that utilizing the $\mathcal{G}_{MST_{CW}}$ graph to compute connectivity constraint sets may leave the network vulnerable to single point failures. We can also notice that from Figure 5.19 that in the underlying heterogenous proximity graph, $\mathcal{G}_{\text{disk}(r(p))}$, there exists redundant links during the simulation. However, from the nature of the dynamic graph, constraints must be imposed on the agents to insure that redundant links exist within the network. One approach to insuring redundant links in the network and therefore network robustness to node failure is computing a *k-vertex-connected* graph and then enforcing this connections through our connectivity constraint sets.

Definition 5.7.1. *(Menger's Vertex Connectivity)[72] A graph is k-vertex-connected if and only if any two distinct vertices are connected by at least k independent paths. Note that two independent paths do not have any internal vertex in common.*

One important property of a *k-vertex-connected* graph is that the graph remains connected when fewer than k vertices are removed [72]. If we can compute a 2-vertex-connected graph, also known as a biconnected graph (G_B), we will maintain network connectivity if one node fails in the network. This redundancy in the network does come at a small cost however. We showed that the fewer links that we take into account when calculating the connectivity constraints lead to larger motion sets for the agents. By enforcing redundancy in the the network we will be shrinking these motion sets.

In the computer science literature computing a biconnected graph is not a new problem. Work has been done on determining approximations to this problem as well as optimal solutions to special cases of the biconnected graph [73]. Although algorithms do exist in the literature, the problem we are looking at may also be considered a special case of the general problem. By default we are computing the MST of our heterogeneous proximity graph, we would like to compute the biconnected graph using the paths already computed by the MST. In other words we would like to compute a second path from each pair of nodes that is independent of the MST paths. Algorithm 2 outlines how to compute a biconnected graph using the MST of a complete graph.

It is straightforward to show that the graph G_B obtained from Algorithm 2 is a biconnected graph. With the construction of the biconnected graph we now have the connections (links) that will insure redundancy in the network. Using the biconnected graph to compute connectivity constraints allows the network to be robust to

Algorithm 2 Biconnected Graph Algorithm (G_B)

```
Assume a complete graph, $G$ with $n + m$ vertices, compute $MST_G$.
Label each vertex from $i = 1, \dots, n + m$.
for each vertex pair $(i, j)$ do
  remove each internal vertex within the $MST_G$ path connecting the vertices and
  replace with the remaining set or subset of vertices.
  if no internal vertex then
    add an internal vertex within the $MST_G$ path.
  end if
  if remaining set of vertices is empty then
    the path is the direct path between $(i, j)$.
  end if
end for
```

single point failures. An example of the MST and a biconnected graph is shown in Figure 5.21 and Figure 5.22 respectively. Algorithm 2 is a general way of obtaining a biconnected graph given the MST of a complete graph and does not address how to compute the minimally weighted biconnected graph. Further investigation is needed to modify the algorithm in order to address this particular problem as well as when the particular graph is not the complete graph.

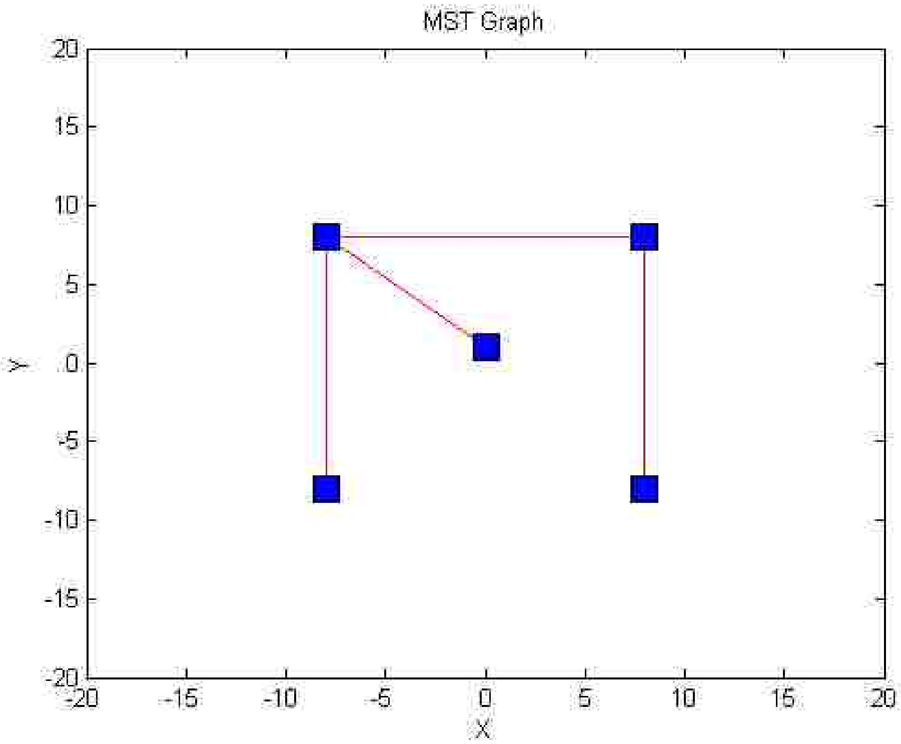

Figure 5.21: Minimum Spanning Tree (MST) of five agents. With one node failure the network will become disconnected.

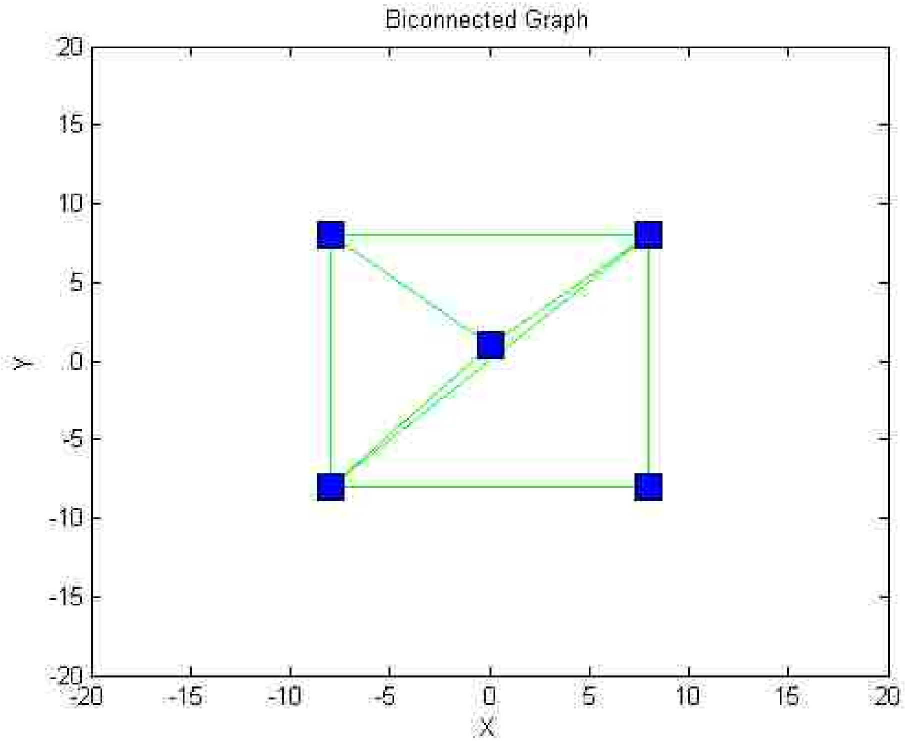

Figure 5.22: Biconnectivity graph of five agents utilizing the MST. Notice that one node failure will not make the network disconnected.

Chapter 6

Prioritized Sensing with Connectivity Constraints

In this chapter we combine our prioritized sensing behavior with the heterogeneous connectivity constraints to create an algorithm that can send sensing agents to areas with the highest possibility of having good information while also guaranteing that the heterogenous network will remain connected. We also evaluate the performance of the heterogeneous algorithm against its homogeneous equivalent to determine the usefulness of heterogeneity within our approach. We also test the prioritized sensing behavior with various number of relays to understand when adding relays to the network begins to show diminishing returns.

6.1 Feasible Motion Sets: Sensing Agents

To combine the prioritized sensing objective with the network connectivity constraints we need to merge our probability of detection map, $(M(q))$, and the connectivity constraint sets we computed in Chapter 5. To do this we refer back to the prioritized sensing algorithm presented in Chapter 3, particularly the set D_i, which is the set of points within robot *i's* Voronoi partition not occupied by obstacles. For ease of notation let us define Υ_i as the connectivity constraint set for robot i as were defined in Section 5.3. Let us now define the feasible motion set, $\mathcal{S}_i$, for robot i as the following,

$$\mathcal{S}_i = D_i \cap \Upsilon_i. \tag{6.1}$$

The feasible motion set $\mathcal{S}_i$ for robot i is the set of points within its Voronoi partition which is not occupied by obstacles and is limited to the points where network communication links can be maintained. Now we have a set, $\mathcal{S}_i$, which we can optimize over that will guarantee network connectivity throughout the search process. The updated prioritized search algorithm which takes into account network connectivity is outlined in Algorithm 3.

In Algorithm 3 it is seen that when network connectivity is taken into consideration, the set over which the probability of detection map is being optimized may "shrink" due to the intersection of the two sets D_i and Υ_i. This makes sense since now the algorithm has to come to a compromise between the sensing objective and the network connectivity objective. Proposition 6.1.1 states a property of the feasible motion set.

Proposition 6.1.1. *Given the set D_i and motion constraint set Υ_i, then the feasible motion set $\mathcal{S}_i \neq \emptyset$.*

Proof. This comes as a consequence of the construction of D_i and Υ_i. Mainly the fact that D_i will always contain at least agent *i's* current position p_i. Also by construction Υ_i will also always contain at least p_i. Therefore in the worst case scenario, $\mathcal{S}_i = D_i \cap \Upsilon_i = p_i$. □

It can be seen by the construction of the feasible motion sets for the sensing

Algorithm 3 Prioritized Sensing with Connectivity Constraints

```
while $t < t_{\text{final}}$ do
  for $x_i = 1, \dots, n$ do
    Calculate $\Upsilon_i$ from $\mathcal{G}_{MST_{CW}}$ (Equations (5.2)-(5.8))
    Determine $V_i \in Q$
    Calculate $\mathcal{S}_i = D_i \cap \Upsilon_i$
    Optimize over $\mathcal{S}_i$ to determine an approximate maximum $\tilde{g}^*$ in $\mathcal{S}_i$ of $M(q)$
    if $\tilde{g}^*$ is reachable then
      $g_{p_i} \Leftarrow \tilde{g}^*$
    else
      $\tilde{g_i}^* = \max(Y_1, \dots, Y_{N-1})$ excluding $\tilde{g}^*$ that was previously calculated
    end if
    Calculate $f_i(p_i, q_{xi}, d)$ in $\mathcal{S}_i$ with $g_{xi}$ set as the goal point.
    $u_i = -k\nabla f_i(\cdot)$ where $k = |p_i - g_{xi}|$
    $\forall q \in R_s$, $M(q) = \beta M(q)$
    Exchange map information with neighbors in $\mathcal{G}_{MST_{CW}}$ graph
  end for
end while
```

agents that taking network connectivity into consideration may produce a suboptimal prioritized search in some cases. This is due to the fact that the set of optimal solutions that guarantee connectivity Υ_i, may not contain the global optimal over an agents particular Voronoi partition. Proposition 6.1.2 states when the algorithm will produce the optimal solution with respect to the prioritized sensing algorithm.

Proposition 6.1.2. *If $\mathcal{S}_i = D_i \cap \Upsilon_i = D_i$ then the goal point g_{p_i} calculated in Algorithm 3 is an optimal solution to the prioritized sensing algorithm presented in Section 3.1.*

Proof. $\mathcal{S}_i = D_i \cap \Upsilon_i = D_i$ implies that $D_i \subset \Upsilon_i$, *i.e.* the set $\mathcal{S}_i$ includes all possible global optima as if connectivity constraints were ignored. Therefore the goal point g_{p_i} is the optimal point in the set D_i which is the optimal solution to the prioritized sensing algorithm outlined in section 3.1. $\square$

Proposition 6.1.2 tells us that if the communication constraints sets are "sufficiently large," then the connectivity constraints can be ignored without losing network connectivity. This is very intuitive in the sense that if an agent can communicate over a larger area than it has been allotted to search, it shouldn't have to consider network connectivity as a constraint on its search/sensing behavior.

6.2 Feasible Motion Sets: Relay Agents

Our approach to addressing connectivity maintenance of a sensor network is to convert the network to a heterogeneous one by adding relay agents capable of better communication capabilities. For this particular application we are assuming that sensing agents are UGVs and relay agents are UAVs flying at a constant altitude. With these assumptions relay agent collisions with sensing agents in the network are not considered. Also, relay agents communication range is considered to be its projection on the two dimensional space.

For the motion planning of relay agents in the network, Algorithm 1 is used to compute the centroid of each relay agents motion constraint set (MCS_i). Therefore, relay agent *i's* feasible motion set is MCS_i. It was observed in section 5.7 that Algorithm 1 produces behavior of the relay agents that acts to balance the network in terms of the distances between its connected links. This is a desirable behavior in sensing/search problems because it allows the network to "stretch" as sensing agents move towards the outer regions of the search space. In essence, as a sensing agent moves towards uninvestigated areas it "pulls" a relay agent with it in order to maintain connectivity of the network.

From algorithm 1 we have the following property which addresses collisions between relay agents at a constant altitude.

Proposition 6.2.1. *Given $MST_{\mathcal{G}_{disk(r(p))}}$ and goal points g_{p_i} computed from Algorithm 1, no two relay agents will occupy the same position at time $kT_s, \forall k \in \mathbb{Z}^+$.*

Proof. Let us assume that agent i and agent j do occupy the position at time kT_s. From Algorithm 1 it seen that this can only occur when agent i and agent j share an edge in the $MST_{\mathcal{G}_{\text{disk}(r(p))}}$ and also share edges with the same agents (vertexes) in the $MST_{\mathcal{G}_{\text{disk}(r(p))}}$. This however is a contradiction of the construction of the minimum spanning tree (MST), *i.e.* the MST is a tree. In other words, links between any two agents (vertexes) are unique. Therefore, no two agents can occupy the same position at time $kT_s, \forall k \in \mathbb{Z}^+$. □

Proposition 6.2.1 states that at each iteration of the algorithm no two relay agents will occupy the same position within the search space. This however is not enough to guarantee collision avoidance for all time t. Since we are assuming a second order system for the agents, we cannot speak to the trajectories between time kT_s and $(k+1)T_s$, *i.e.,* the agents may experience overshoot, steady state error, etc. However, with a well tuned system, collisions between relay agents can be addressed during their arrivals to their respective goal points.

This leads us to the final property when Algorithm 1 and Algorithm 3 are utilized to control a heterogenous sensor network composed of relay (UAVs) and sensing (UGVs) agents.

Theorem 6.2.2. *Given a heterogeneous sensor network as described in Section 5.2 where relay agents utilize inputs described by Algorithm 1 and sensing agents utilize* *inputs described in Algorithm 3, then the heterogeneous sensor network will remain connected and collision free throughout the search process.*

Proof. Collision avoidance for the sensing agents within the network is guaranteed from Proposition 3.2.1. Also, since we are assuming that relay agents are UAVs flying above sensing agents (UGVs), sensor and relay agent collisions are avoided. Lastly, collisions avoidance between relay agents is guaranteed by Proposition 6.2.1. Connectivity maintenance is guaranteed by Theorems 5.6.4 and Theorem 5.6.5. □

Remark 6.2.3. *It is also key to notice that by construction Algorithm 1 and Algorithm 3 can be implemented in a decentralized fashion. With an initially connected network, the Voronoi partitions can be constructed with only neighboring agents knowledge. Under the assumption that there exist synchronization within the network and all agents can broadcast position information with a unique id, then the heterogeneous proximity graph as well as the MST of the heterogeneous proximity graph can be constructed in a decentralized fashion [71].*

6.3 Simulations

For the following simulations relay agents apply Algorithm 1 while sensing agents apply Algorithm 3 . To understand how the communication constraints as well as adding relay agents to the sensor network affects the prioritized sensing behavior, we simulate two scenarios. For the first simulation we implement Algorithm 3 on a heterogeneous network made up of 7 sensing agents and 4 relay agents. The search space is taken to be $60m \times 60m$ square area, with the communication ranges for the sensing agents, $R_c = 3m$ and $R_{rc} = 16m$ for the relay agents. The sensing agents in the network are initialized in a random configuration with the relay agents situated at (-8m,-8m),(-8m,8m),(8m,-8m),(8m,8m) such that the initial configuration of the

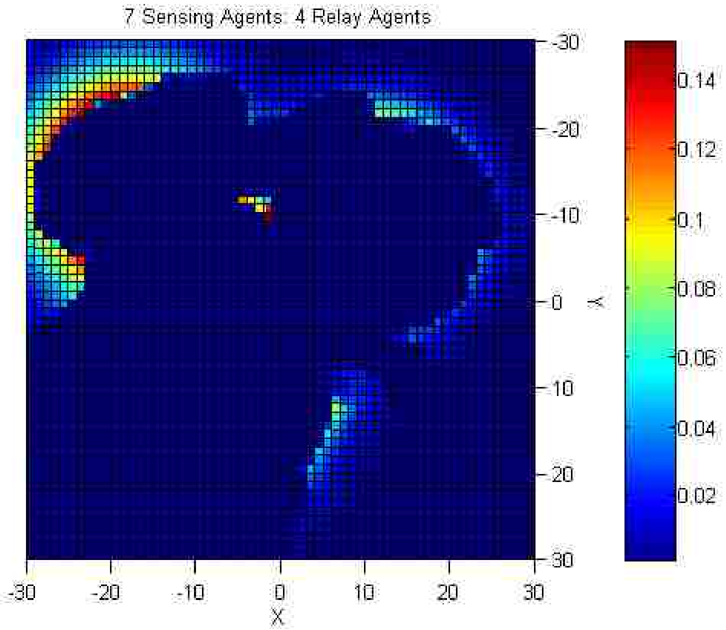

Figure 6.1: Probability of Detection Map after 50 iterations of the algorithm. Notice that most of the area has been searched with exception to the upper left hand corner.

heterogeneous network is connected. The sensing radius of the sensing agents are taken to be $3m$ and the parameter that reflects the reduction of the probability of detection map, β is taken to be 0.8. Each simulation lasts for 50 iterations of the algorithm, approximately 120 seconds. Table 6.1 summarizes the results for five simulations.

Table 6.1: Heterogeneous Sensor Network (7 sensing, 4 relay agents): Prioritized Sensing

Simulation Number	Average POD per m^2	Maximum POD Value
1	0.0043	0.178
2	0.0104	0.337
3	0.0034	0.168
4	0.0031	0.124
5	0.0147	0.333

Figure 6.1 shows the reduced POD map, $M(q)$, after 50 iterations of the algorithm with seven sensing agents and four relay agents. Notice that most of the area has already been searched and has been reduced to around 0.14.

In the second set of simulations we assume there are no relay agents in the network. All parameters were kept the same as in the first set of simulations except only 7 sensing agents were used in the sensor network. This set of simulations shows how the prioritized sensing algorithm performs when the communication constraints are imposed on the homogeneous network of only sensing agents without the help of specialized relay agents. The results of this set of simulations are summarized in Table 6.2.

Table 6.2: Homogeneous Sensor Network (7 sensing agents): Prioritized Sensing

Simulation Number	Average POD per m^2	Maximum POD Value
1	0.143	0.98
2	0.144	0.99
3	0.144	0.98
4	0.142	0.97
5	0.146	0.99

Figure 6.2 shows the reduction of the POD map after 50 iterations when the connectivity constraints are imposed directly on the homogeneous sensor network of seven sensing agents. Notice that the majority of the area has yet to be searched and the most probable areas of containing good information have not been searched. This shows how much the connectivity constraints of the network can inhibit an efficient search of the area.

From Table 6.2 we see that imposing the communication constraints directly on the network with only sensing agents hinders the prioritized sensing algorithm significantly. The main reason for such poor results as compared to the heterogeneous network with relays (Table 6.1) is the fact the communication radius of each sensing

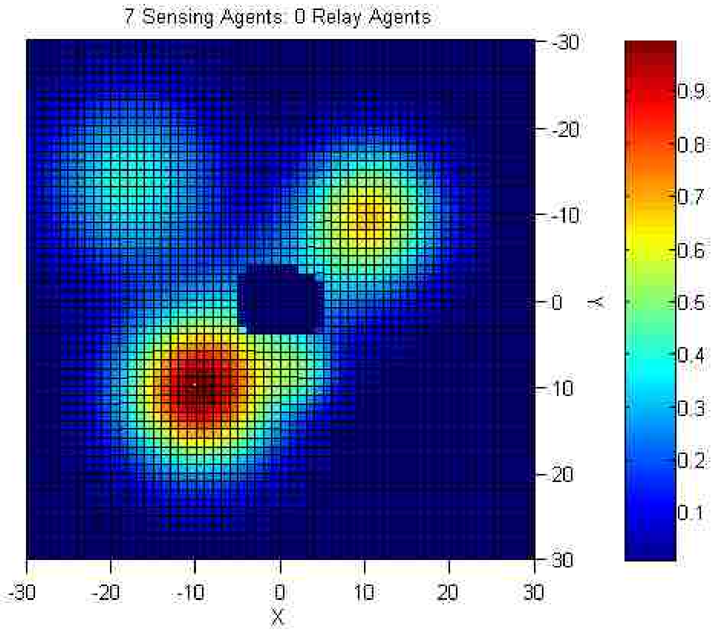

Figure 6.2: Probability of Detection Map after 50 iterations of the algorithm with a homogeneous sensor network with connectivity constraints. Notice that most of the area has not been searched within the time allotted.

agent is very small, $3m$, as compared to the search area they are tasked to explore, $60m \times 60m$. From the first two sets of simulations it is clear that utilizing a heterogeneous sensor network is advantages when the area to be scanned is much larger than that of the communication radius of the sensing agents. As we first hypothesized, the relay agents enable the sensor network to have a larger reach, thus enabling a more efficient behavior of the prioritized sensing algorithm when communication constraints are taken into consideration.

The next set of simulations we look at is when relay agents are replaced by more sensing agents. In this scenario the sensor network is comprised of 15 sensing agents and zero relay agents. This set of simulations should give us an indication if simply adding more sensing agents to the network can overcome the constriction of the network constraints. Table 6.3 shows the results for this particular scenario.

Table 6.3: Large Homogeneous Sensor Network (15 sensing agents): Prioritized Sensing

Simulation Number	Average POD per m^2	Maximum POD Value
1	0.128	0.98
2	0.125	0.97
3	0.122	0.97
4	0.124	0.98
5	0.131	0.98

Comparing Table 6.1 with Table 6.3 we see that even with more than twice the amount of sensing agents in the network compared to the previous simulations, the communication constraints do not allow the network to “stretch out” enough to efficiently search the area in the time allotted. This clearly shows the advantage of using specialized agents, relays in this case, to help extend the range of the sensor network.

To understand the effect of adding relay agents to the sensor network we conduct several simulations of the algorithm with different numbers of relay agents for 50 iterations, approximately 120 seconds. Table 6.4 summarizes the outcome of these particular simulations. Note that when 16 relay agents are used in the sensor network, complete communication coverage of the search area is obtained.

Figure 6.3 shows a graph of how the maximum POD value after the algorithm was run for 50 iterations changed with additional relay agents in the network. We can see a fairly good improvement of the maximum POD value with the addition of up to 8 relay agents. In these simulations, adding more than 8 relay agents to the network did not significantly change the outcome of the algorithm in the allotted time.

Table 6.4: Adding Relay Agents to Sensor Network (15 sensing agents): Prioritized Sensing

Number of Relay Agents	Average POD per m^2	Maximum POD Value
1	0.044	0.504
2	0.031	0.396
3	0.018	0.387
4	0.002	0.174
5	0.002	0.161
8	7.9×10^{-5}	0.020
10	7.2×10^{-5}	0.009
16(Full Coverage)	6.3×10^{-7}	0.001

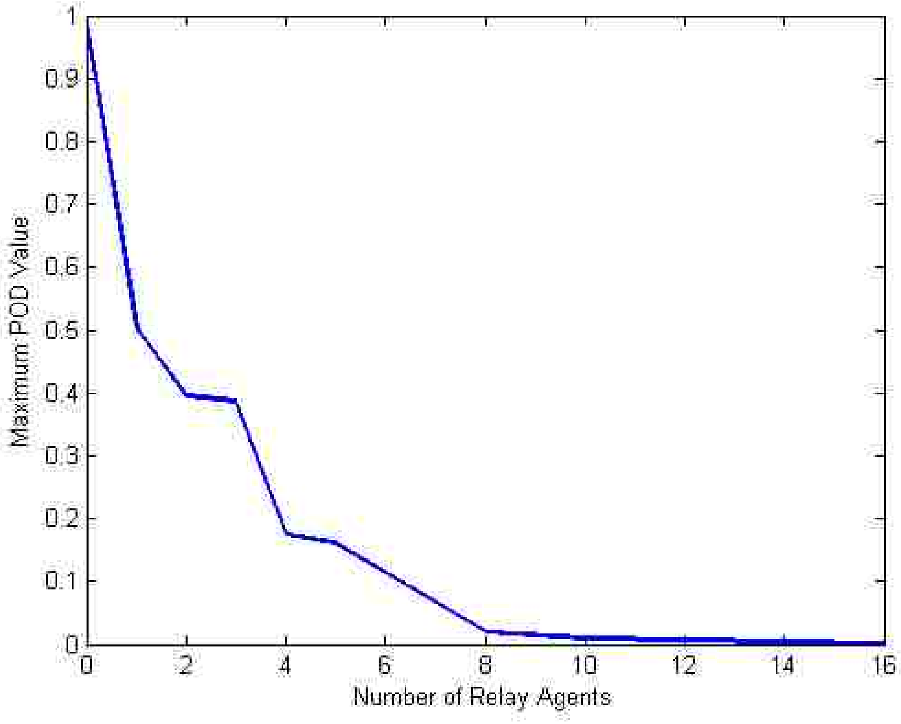

Figure 6.3: Change in maximum POD map value with additional relay agents in the sensor network. The simulations lasted 50 iterations of the algorithm or about 120 seconds.

Part III: Experimental Validation

Chapter 7

Multi-Vehicle Testbed for Decentralized Environmental Sensing

In this chapter we present our multi-vehicle testbed that was designed for verification and validation of cooperative control algorithms involving environmental sensing [74]. The multi-vehicle testbed allows for a straightforward transition from simulation to experimenting on actual hardware and has the flexibility to interface various types of sensors, vehicles, as well as enable indoor and outdoor experiments.

7.1 Introduction

Recently in the literature much attention has been paid to the development of mobile robot teams capable of accomplishing various tasks through cooperation which would be very inefficient if done by a single robot. Among the recent advances in mobile robots is the idea of a dynamic or reconfigurable sensor network, where each robot is equipped with sensors that are capable of measuring some parameter of the environment and able to reconfigure the network configuration based on these measurements. Through cooperation the robot team should accomplish different tasks such as optimal sensor coverage, target tracking, or spatial distribution mapping. Some motivating and practical applications include search and rescue operations [3], [75], target detection [76], [4], and hazardous contaminations [5] to name a few.

Although much attention has been paid to creating cooperative control algorithms for dynamic sensor networks less attention has been paid to the validation and verification of these control algorithms on experimental hardware. The focus of this paper is to expand our current experimental testbed to accommodate environmental sensing applications.

To facilitate the development of novel cooperative control algorithms specifically for environmental sensing, monitoring, and mapping, we extend our current multi-vehicle testbed to enable a quick turnaround from simulations of the control algorithms to actual hardware implementation.

With this addition, limitations in the cooperative control algorithms can be identified and subsequently addressed in further research. This approach to verification and validation facilitates cooperative control algorithms that are implementable in real-wold scenarios.

The addition to our multi-vehicle testbed consists of four Pioneer P3-AT mobile robots each equipped with an environmental sensor suite. The multi-vehicle testbed was designed to interface various types of sensors as well as various numbers of vehicles and types. Sensors and robots can be added and removed on the fly based on user need. Also, the multi-vehicle testbed is able to carry out indoor as well as outdoor experiments.

7.1.1 Related Experimental Testbeds

We expand on our original testbed [77] COMET, to include mechanisms that allow for environmental sensing which enable validation and verification of cooperative control algorithms that depend on measurements of the sensed environment. The original COMET testbed consisted of ten all-terrain vehicles which are based on the Tamiya TXT-1 chasis. The COMET testbed is used for validation of cooperative control algorithms, however in its first generation lacked environmental sensing capabilities.

Along the same lines, the experimental testbed at the GRASP Laboratory at the University of Pennsylvania [78] was designed for large-scale multi-robot systems for experimental validation of distributed robot applications in a strictly indoor environment. This testbed was specifically designed to address situations such as formation control, search and pursuit of targets of interest, and cooperative manipulation tasks.

In a similar design to the testbed presented in this dissertation, the authors of [79] utilize a group of 16 SwarmBots to validate a coverage control algorithm that is based upon information of the sensed environment. Sensory information was simulated during one of the experiments to compare the performance against a known ground truth then during a second experiment sensory information was taken from onboard light sensors.

The GRASP testbed is more tailored for validating control algorithms such as formation control, search and pursuit of targets, and cooperative manipulation, rather than sensing environmental data which is the focus of our testbed. Also, the group of SwarmBots use only a single source of data rather than data from multiple spatial distributions in the environment which our current testbed is equipped to handle.

7.2 Hardware Description

7.2.1 Vehicle Description

The Pioneer P3-AT stores up to 252 Wh of hot-swappable batteries. The P3-AT can reach speeds of 0.8 m/s and carry a payload of up to 30 kg as well as climb a steep 45% gradient. Also, laser-based navigation options, integrated inertial correction to compensate for slippage, GPS, bumpers, gripper, vision, stereo rangefinders, and compass options are available commercially for the P3-AT [80].

Our current testbed can accommodate laser-based navigation, GPS navigation, as well as gripper/manipulator tasks. Although this paper concentrates on experiments conducted with the Pioneer P3-AT robots, our testbed also contains ten all-terrain vehicles which are based on the Tamiya TXT-1 chasis as well as a Drangonflyer X-Pro quadrotor and two AscTec Hummingbird quadrotors.

7.2.2 Environmental Sensor Suite

The environmental sensor suite consists of a Phidgets 8/8/8 USB interface I/O board capable of measuring eight digital and eight analog inputs and capable of driving eight digital outputs. The Phidgets I/O board can accommodate pressure, temperature, humidity, light intensity, and magnetic field sensors as well as many others.

A special aluminium plate and mounting system was created to interface the environmental sensor suite. The plate and mounting system allows for multiple sensor configurations as well as the ability to mount multiple accessories on each robotic platform. Figure 7.1 shows the custom built aluminum plate with four precision light sensors and three magnetic sensors. Also shown is a Hokuyo UHG-08LX laser range finder. The addition of the custom plate and mounting brackets allow for a quick

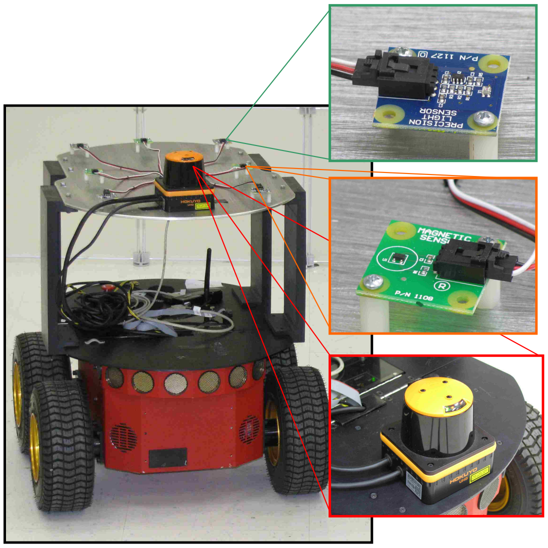

Figure 7.1: Pictured are four precision light sensors (top) three magnetic sensors (middle) and a Hokuyo UHG-08LX laser range finder (bottom) all mounted on a custom fixture that is attached to a robot.

swapping of sensors and accessories to address a variety of experimental tests.

7.3 Player/Stage/Gazebo/USARSim Interface

One of the most widely used robotics software packages is the Player/Stage/Gazebo (PSG) [81]. The PSG project consists of libraries that provide access to communication and interface functionality on robot hardware. The robot "server" Player, provides an architecture where multiple modules, also known as drivers, can be written independently and connected via a custom middleware that relies on TCP communication. Users write "client" applications (control algorithms) that connect to and command modules (drivers) running on a Player "server." Additionally, PSG provides a 2D simulator, Stage, and a 3D physics-based simulation environment Gazebo. Additionally USARSim [82] is a high-fidelity simulator of robots as well as environments which is based upon the Unreal Tournament game engine which can be used with a Player “server.” USARSim allows for realistic robotic environments with kinematically accurate robot models. These simulators provide a transparent transition from simulation code using a virtual environment to the actual robot hardware.

Chapter 8

Prioritized Sensor Detection: Homogeneous Sensor Network

In this chapter we study the results from an experiment conducted to validate the prioritized sensor detection algorithm found in Section 3.1. The algorithm was conducted on a network of homogeneous mobile sensors where connectivity between sensors was assumed in the network was assumed.

8.1 Experimental Results

Experimental tests were conducted at the Marhes laboratory located at the University of New Mexico. Three Pioneer P3-AT mobile robots were each equipped with a Hokuyo UHG-08LX laser range finder capable of a 240 degree scanning area used for obstacle avoidance as well as an environmental sensor suite used for mapping spacial distributions. Each sensor suite consists of a Phidgets USB 8/8/8 interface kit that enables measurements from eight analog inputs and eight digital inputs as well as the ability to drive eight digital outputs. Four precision light sensors capable of measuring a range of 1 lux to 1000 lux was used to create a spatial distribution mapping of light intensity. Three magnetic sensors were also used in the multi-sensing experiment to map magnetic intensities.

Figure 8.1 depicts the experimental setup of our environmental testbed for the multi-sensing experiment. The two spatial distribution quantities that our sensors are measuring are magnetic intensity as well as light intensity. In Figure 8.1 we see that a magnetic source (large magnet) is placed at $(0.5\text{m}, -1.5\text{m})$ and a light source is emanating directly above $(0\text{m}, 2\text{m})$ in the search space. The magnetic source used during the experiment was a "C" shaped ferrite magnet placed in a box, which is capable of producing a magnetic intensity of 126 gauss at a distance of 40cm. The light source used for the experiment was a 100watt flood lamp placed 2.5m above the ground. Also two obstacles were placed in the search area to show the collision avoidance capability of the control algorithm.

Figure 8.2 shows the initial POD map used for the experiment. Each robot first calculates its own Voronoi partition based on its position as well as its neighbors. Next the point that has the highest probability of containing "good" information is calculated. For our experiment "good" information represents a magnetic or light source. We notice in Figure 8.2 that after 30 iterations of the algorithm the POD map has been reduced by a factor of two. Figure 8.4 shows the points in the search space that the robots took measurements from, in other words, where the highest probability of obtaining good magnetic or light readings.

Figure 8.3 depicts the magnetic and light intensity maps after 30 iterations of the control algorithm. We see that indeed the highest intensity values coincide with locations of the magnetic and light sources respectively. We also notice that two of the robots were taking measurements that coincided with the magnetic and light sources, however the third robot was taking measurement that did not coincide with a source. This shows the dependence of the control algorithm on prior information.

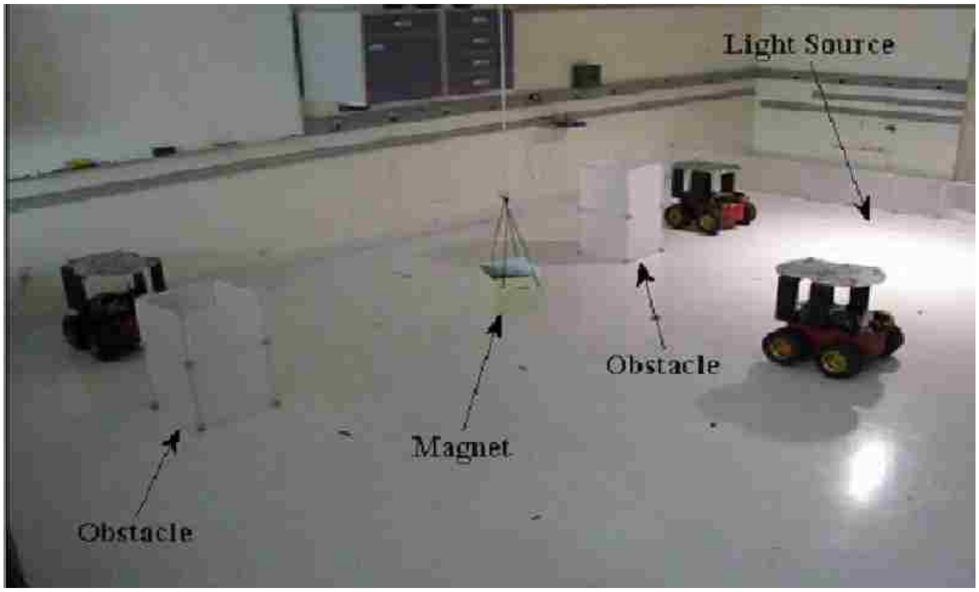

Figure 8.1: The experimental setup for the prioritized multi-sensing control algorithm. Notice the magnetic source, light source, and obstacles.

Although the experiment was only ran for 30 iterations, in simulation it is observed that the third robot would eventually make its way towards one of the other robots to help reduce the POD map or begin reducing the POD map in areas that have yet to be reduced.

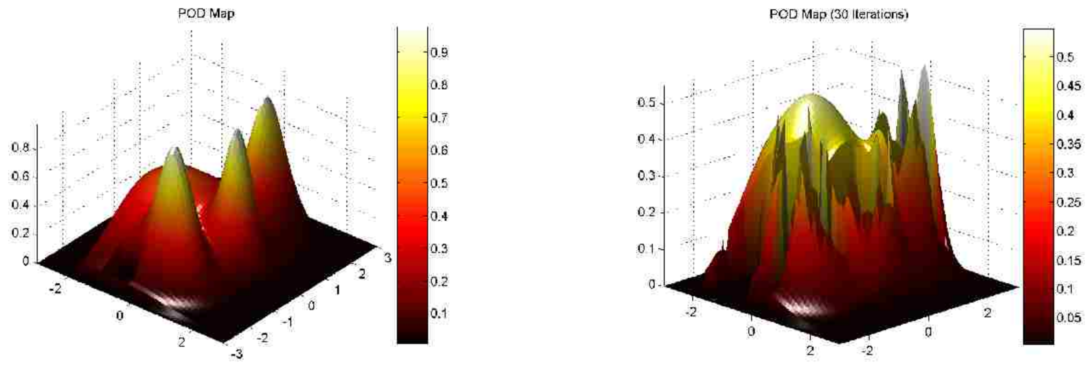

Figure 8.2: The figure on the left shows the probability of detection map $M(q)$ which reflects the likelihood of detecting "good" information in the region. The figure on the right shows the probability of detection map after 30 iterations of the control algorithm for the multi-sensing behavior. Notice that the probability of detection has been reduced significantly, from about 0.95 to nearly 0.5.

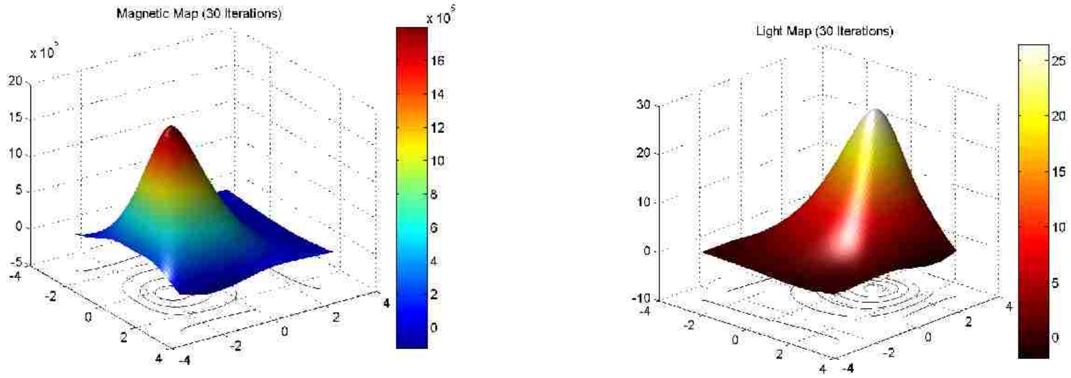

Figure 8.3: The figure on the left shows the magnetic intensity map after 30 iterations of the cooperative control algorithm. We see that the highest concentration of the magnetic field is near (.5m,-1.5m). The figure on the right shows the light intensity map after 30 iterations of the cooperative control algorithm. We see that the light is concentrated near (0m,2m).

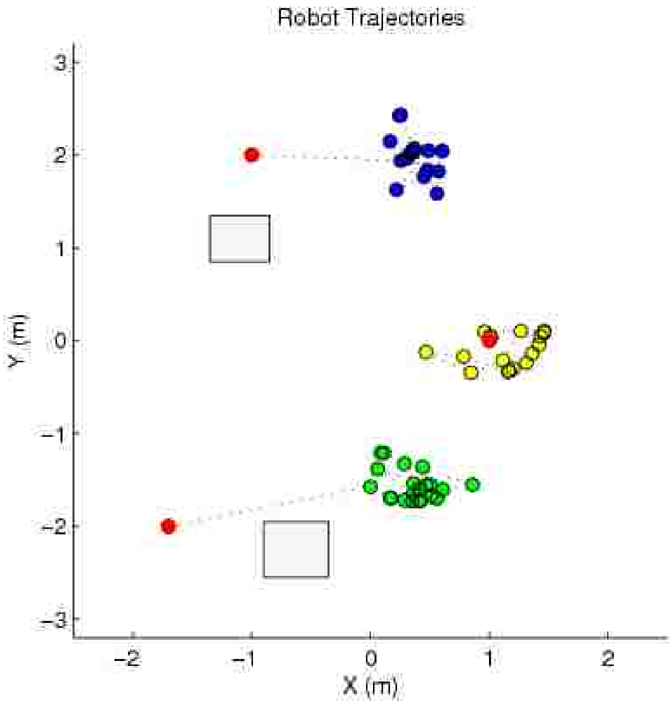

Figure 8.4: The robot trajectories during the multi-sensing experiment. The red dots correspond with the robots initial positions and the yellow, blue, and green dots represent the goal points the robots navigated to.

Chapter 9

Centroidal Motion Constraint Set Configurations: Heterogeneous Sensor Network

In this chapter we implement the centroidal heterogeneous motion constraint set configurations presented in Section 5.7. This hardware experiment was conducted to validate the claims of Theorem 5.6.4 and Theorem 5.6.5.

9.1 Experimental Results

This section looks at the particular situation when the relay agent of the heterogeneous team moves towards its centroid of its feasible motion set. The rest of the heterogeneous network consists of two sensing agents. In this particular network configuration the relay robots behavior mimics that of balancing the network in the form of keeping equidistance between the two sensing agents. In this sense, the relay agent is trying to give equal network considerations to each sensing agent.

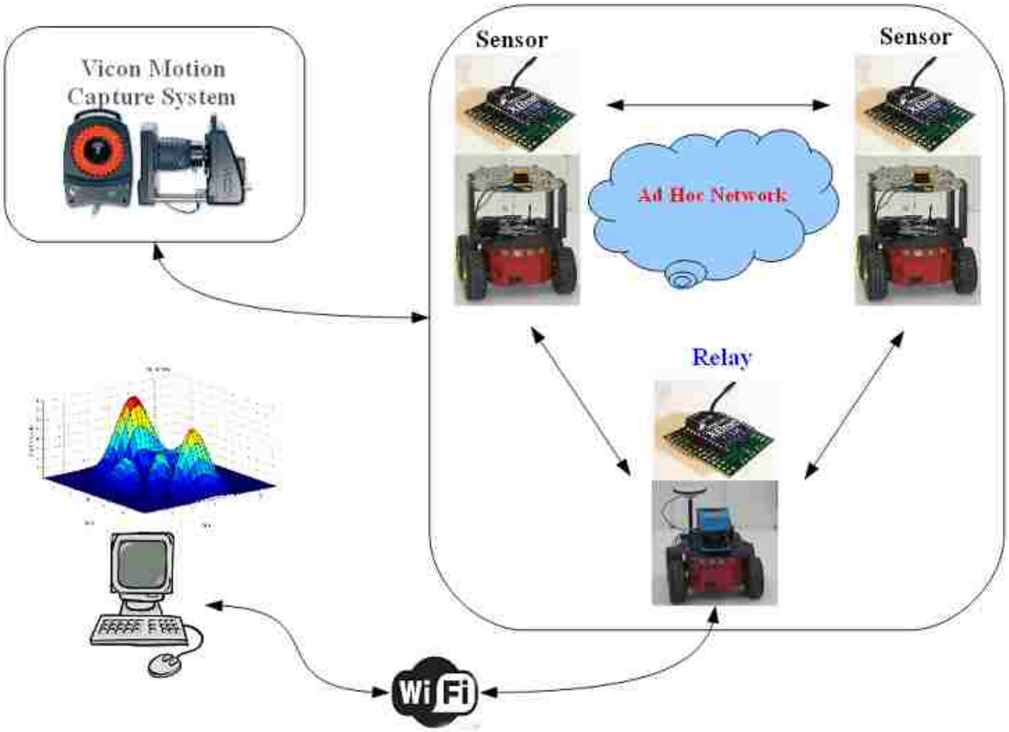

Figure 9.1: Diagram of hardware experiments using the centroid seeking behavior.

For the hardware implementation of the communication constraints, we chose to implement Algorithm 1 on a single relay robot. Two sensing robots were given predefined trajectories and were tasked with taking light intensity measurements along these trajectories. The relay robot calculates its feasible motion set based on the positions of the sensing agents and then moves towards its centroid. Position information of the sensing agents were updated every 0.5 seconds. For this experiment we used $R_{rc} = 3.2$m and $R_c = 1.0$m for the communication radius of the relay and sensing agents respectively. Figure 9.1 shows a diagram of how the experiment was implemented and Figure 9.2 shows a snapshot of the hardware setup. The ad-hoc network consisting of three XBee wireless RF Modules was used to communicate sensing data between robots. This allowed for the light intensity map to be built

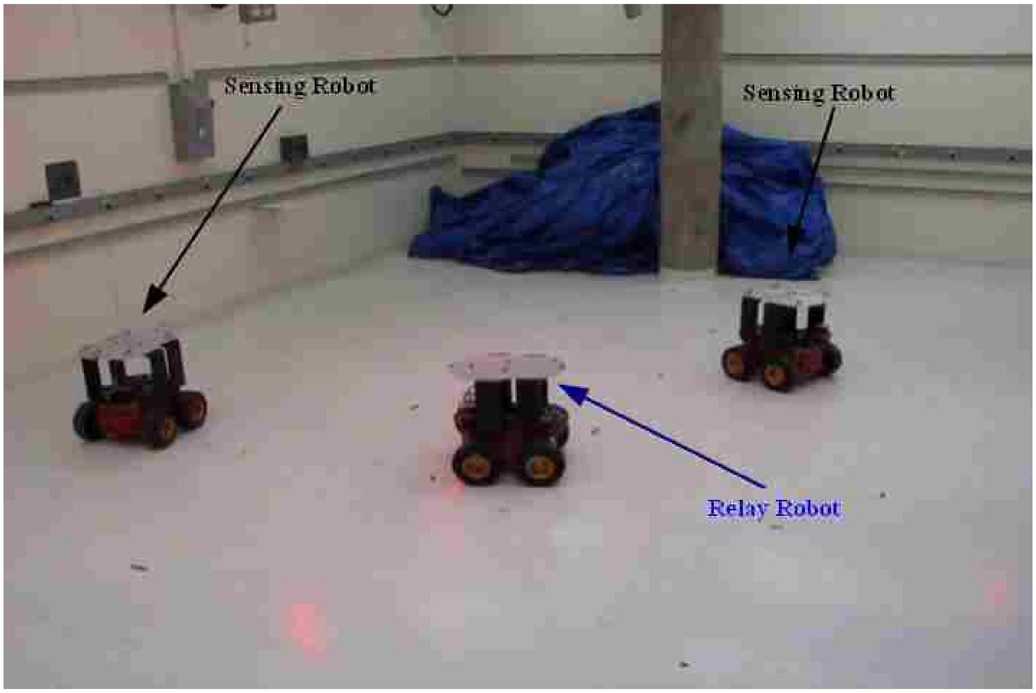

Figure 9.2: Experimental snapshot showing the two sensing agents and a single relay agent.

in a distributed fashion. A wireless local area network (WLAN) was used to send a real-time light intensity map from the relay robot to an end user using a laptop outside the experimental area. Figure 9.3 shows the evolution of the second smallest eigenvalue of the $\mathcal{G}_{\text{disk}(r(p))}$ graph and shows that the heterogeneous network stays connected for the entire experiment.

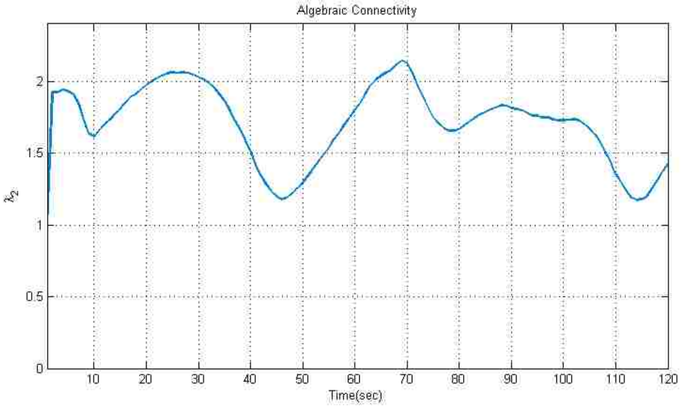

Figure 9.3: The second smallest eigenvalue of the $\mathcal{G}_{\mathrm{disk}(r(p))}$ graph during experiments of the centroidal behavior. The network remains connected throughout the experiment.

Chapter 10

Conclusion and Future Work

In this dissertation we addressed prioritized sensing behaviors with communication constraints for a heterogeneous sensor network made up of sensing agents and mobile communication relays. First we provided a collision free motion controller that drives sensing agents to areas within their search area that contain the highest probability of containing "good information." We also provided a technique for combining different sensing objectives which relied on logistic regression. Secondly, we derived connectivity constraints for a heterogeneous sensor network which allowed for the development of feasible motion sets that guarantee network connectivity for agents within the network. Lastly, we showed how to reduce the number of communication constraints to allow the sensing agents to maximize their feasible motion sets and thus allow for a larger search area while maintaining network connectivity. A technique for shaping the network configuration was also presented that allows for biasing particular communication links within the network which shapes the flow of information within the sensor network.

Future research directions include formulating network connectivity in a probabilistic sense, *i.e.*, assign a probability of becoming disconnected given certain configurations rather than a strick geometric approach to connectivity maintenance. Another area for extending this framework lies in relaxing the assumption that the sending and receiving range of each agent is symmetric. One question that can be asked is, how does having a larger sending range than receiving range change the heterogeneous proximity graph construction and what are the implications to the efficiency of the prioritized search? Also a more in depth investigation is needed to understand the robustness of the network to node failures and how it may be incorporated in our current framework.

References

[1] J. C. Latombe, *Robot Motion Planning.* Kluwer, 1991.

[2] J. Cortés, S. Martínez, and F. Bullo, "Spatially-distributed converage optimization and control with limited-range interactions," *ESIAM: Control, Optimization and Calculus of Variations*, vol. 11, no. 4, pp. 691–719, 2005.

[3] B. Lavis, Y. Yokokohji, and T. Furukawa, "Estimation and control for cooperative autonomous searching in crowded urban emergencies," *IEEE International Conference on Robotics and Automation*, pp. 2831–2836, May 2008.

[4] 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 Optimization*, vol. 48, no. 1, pp. 292–320, 2009.

[5] R. Cortez, X. Papageorgiou, H. G. Tanner, A. V. Klimenko, K. N. Borozdin, R. Lumia, and W. C. Priedhorsky, "Smart radiation sensor management: Nuclear search and mapping using mobile robots," *IEEE Robotics and Automation Magazine*, vol. 15, no. 3, pp. 85–93, September 2008.

[6] M. Huntwork, A. Goradia, N. Xi, C. Haffner, C. Klochko, and M. Mutka, "Pervasive surveillance using a cooperative mobile sensor network," in *IEEE International Conference on Robotics and Automation*, May 2006, pp. 2099–2104.

[7] A. Singh, R. Nowak, and P. Ramanathan, "Active learning for adaptive sensing newtorks," in *5th International Conference on Information Processing in Sensor Networks*, 2006, pp. 60–68.

[8] S. Oh, P. Chen, M. Manzo, and S. Sastry, "Instumenting wireless sensor networks for real-time surveillance," in *IEEE International Conference on Robotics and Automation*, May 2006, pp. 3128–3111.

[9] S. Martínez, J. Cortés, and F. Bullo, "Motion coordination with distributed information," *IEEE Control Systems Magazine*, vol. 27, no. 4, pp. 75–88, August 2007.

[10] I. I. Hussein and D. M. Stipanovic, "Effective coverage control for mobile sensor networks," in *IEEE Conference on Decision and Control*, 2006, pp. 2747–2752.

[11] R. A. Cortez and H. G. Tanner, "Radiation mapping using multiple robots," *2nd ANS International Joint Topical Meeting on Emergency Preparedness and Response and Robotic and Remote Systems*, pp. 157–159, March 2008.

[12] L. C. Pimenta, V. Kumar, R. C. Mesquita, and G. A. S. Pereira, "Sensing and coverage for a network of heterogeneous robots," *IEEE Conference on Decision and Control*, pp. 3947–3852, December 2008.

[13] M. ling Lam and Y. hui Liu, "Two distributed algorithms for heterogeneous sensor network deployment towards maximum coverage," *IEEE International Conference on Robotics and Automation*, pp. 3296–3301, May 2008.

[14] L. Lazos and R. Poovendran, "Coverage in heterogeneous sensor networks," in *4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, April 2006, pp. 1–10.

[15] H. G. Tanner and D. K. Christodoulakis, "Decentralized cooperative control of heterogeneous vehicle groups," *Robotics and Autonomous Systems*, vol. 55, no. 11, pp. 811–823, 2007.

[16] M. Kumar, D. Garg, and V. Kumar, "Self-sorting in a swarm of heterogeneous agents," *American Control Conference*, pp. 117–122, June 2008.

[17] L. E. Parker, B. Kannan, X. Fu, and Y. Tang, "Heterogeneous mobile sensor net deployment using robot herding and line-of-sight formations," in *Proc. of the 2003 IEEE/RSJ Int. Conference of Intelligent Robots and Systems*, Las Vegas, NV, 2003, pp. 2488–2493.

[18] L. E. Parker, B. Kannan, F. Tang, and M. Bailey, "Tightly-coupled navigation assistance in heterogeneous multi-robot teams," in *Proc. of the 2004 IEEE/RSJ Int. Conference of Intelligent Robots and Systems*, Sendai, Japan, 2004, pp. 1016–1022.

[19] J. Fax and R. Murray, "Graph laplacian and stabilization of vehicle formations," *Proceeding of the 15th IFAC*, pp. 283–288, 2002.

[20] H. G. Tanner, A. Jadbabaie, and G. Pappas, "Stable flocking of mobile agents, part 1: Fixed topology," *IEEE Conference on Decision and Control*, pp. 2010–2015, December 2003.

[21] D. V. Dimarogonas and K. H. Johansson, "Decentralized connectivity maintenance in mobile networks with bounded inputs," *IEEE International Conference on Robotics and Automation*, pp. 1507–1512, May 2008.

[22] A. Muhammad, M. Ji, and M. Egerstedt, "Applications of connectivity graph processes in networked sensing and control."

[23] N. Ayanian, V. Kumar, and D. Koditschek, "Synthesis of controllers to create, maintain, and reconfigure robot formations with communication constraints," in *Proceedings of International Symposium of Robotics Research*, Luzern, Switzerland, August 2009.

[24] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, "Maintaining connectivity in mobile robot networks," *O. Khatib et. al (EDS.): Experimental Robotics: The 11th International Symposium., STAR 54*, pp. 117–126, 2008.

[25] M. M. Zavlanos and G. J. Pappas, "Distributed connectivity control of mobile networks," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1416–1428, December 2008.

[26] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph laplacian," *IEEE Transactions on Automatic Control*, vol. 51, no. 1, pp. 116–120, 2006.

[27] M. Ji and M. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness," *IEEE Transactions On Robotics*, vol. 23, no. 4, pp. 693–703, August 2007.

[28] D. P. Spanos and R. M. Murray, "Robust connectivity of networked vehicles," *IEEE Conference on Decision and Control*, pp. 2893–2898, December 2004.

[29] J. Fink and V. Kumar, "Online methods for radio signal mapping with mobile robots," *IEEE International Conference on Robotics and Automation*, pp. 1940–1945, May 2010.

[30] O. Tekdas, P. A. Plonski, N. Karnad, and V. Isler, "Maintaining connectivity in environments with obstacles," *IEEE International Conference on Robotics and Automation*, pp. 1952–1957, May 2010.

[31] O. Tekdas, W. Wang, and V. Isler, "Robotic routers: Algorithms and implementation," *The International Journal of Robotics Research*, vol. 29, no. 1, pp. 110–126, January 2010.

[32] O. Burdakov, P. Doherty, K. Holmberg, J. Kvarnstrom, and P. R. Olson, "Positioning unmanned aerial vehicles as communication relays for surveillance tasks," in *Conference on Robotics Science and Systems*, 2009.

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

[34] G. Hollinger and S. Singh, "Multi-robot coordination with periodic connectivity," *IEEE International Conference on Robotics and Automation*, pp. 4457–4462, May 2010.

[35] N. Chakraborty and K. Sycara, "Reconfiguration algorithms for mobile robotic networks," *IEEE International Conference on Robotics and Automation*, pp. 5484–5489, May 2010.

[36] E. W. Frew, "Information-theoretic integration of sensing and communication for active robot networks," *Mobile Networks and Applications*, vol. 14, no. 3, pp. 267–280, June 2009.

[37] I. Hussein, D. Stipanović, and Y. Wang, "Reliable coverage control using heterogeneous vehicles," in *IEEE Conference on Decision and Control*, New Orleans, 2007, pp. 6142–6147.

[38] M. Stachura and E. W. Frew, "Reliable coverage control using heterogeneous vehicles," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Canada, August 2010.

[39] H. G. Tanner, "Switched UAV-UGV cooperation scheme for target detection," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3457–3462.

[40] R. Y. Rubinstein, *Simulation and The Monte Carlo Method.* John Wiley and Sons, 1981.

[41] R. A. Cortez, R. Fierro, and J. Wood, "Prioritized sensor detection via dynamic voronoi-based navigation," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2009, 5815-5820.

[42] A. Torn and A. Zilinskas, *Global Optimization.* Springer-Verlag, 1987.

[43] C. Belta and V. Kumar, “Abstractions and control policies for a swarm of robots,” *IEEE Transactions on Robotics*, vol. 20, no. 5, pp. 865–875, 2004.

[44] O. A. A. Orqueda and R. Fierro, “Visual tracking of mobile robots in formation,” *American Control Conference*, pp. 5940–5945, July 2007.

[45] J. Lin, A. S. Morse, and B. D. O. Anderson, “The multi-agent rendezvous problem,” *IEEE Conference on Decision and Control*, pp. 1508–1513, December 2003.

[46] J. Cortés, S. Martínez, and F. Bullo, “Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions,” *IEEE Transactions on Automatic Control*, vol. 51, no. 8, August.

[47] D. Koditscheck and E. Rimon, “Robot navigation functions on manifolds with boundary,” *Advances in Applied Mathematics*, vol. 11, pp. 412–442, 1990.

[48] H. G. Tanner and A. Kumar, “Towards decentralization of multi-robot navigation functions,” in *IEEE International Conference on Robotics and Automation*, 2005, pp. 4132–4137.

[49] S. G. Loizou and A. Jadbabaie, “Density functions for navigation-function-based systems,” *IEEE Transactions on Automatic Control*, vol. 53, no. 2, pp. 612–617, March 2008.

[50] A. Ghaffarkhah and Y. Mostofi, “Communication-aware target tracking using navigation functions - centralized case,” *International Conference on Robot Communication and Coordination (ROBOCOMM)*, pp. 1–8, 2009.

[51] P. Ogren and N. Leonard, “A convergent dynamic window approach to obstacle avoidance,” *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 188–195, April 2005.

[52] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerishce Mahematik 1*, pp. 269–271, 1959.

[53] B. Bollobás, *Modern Graph Theory.* Springer-Verlag New York Inc., 1998.

[54] P. Yang, R. A. Freeman, and K. M. Lynch, “Distributed cooperative sensing using consensus filters,” *IEEE International Conference on Robotics and Automation*, pp. 405–410, April 2007.

[55] A. K. Das, R. Fierro, V. Kumar, J. P. Ostrowski, J. Spletzer, and C. J. Taylor, “A vision-based formation control framework,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 813–825, 2002.

[56] Y. Yuan and H. Tanner, "Sensor graphs for guaranteed cooperative localization performance," *Control and Intelligent Systems*, 2010, to appear.

[57] R. Cortez, D. T. I. Palunko, R. Fierro, and J. Wood, *Intelligent Systems Book for AIAA Progress in Aeronautics & Astronautics Series*, 2010, accepted.

[58] B. Lavis, T. Furukawa, and H. F. Durrant-Whyte, "Dynamic space reconfiguration for Bayesian search and tracking with moving targets," *Auton. Robots*, vol. 24, no. 4, pp. 387–399, 2008.

[59] J. Tisdale, Z. Kim, and K. Hedric, "Autonomous UAV path planning and estimation," *IEEE Robotics and Automation Magazine*, vol. 16, no. 2, pp. 35–42, 2009.

[60] G. M. Hoffmann and C. J. Tomlin, "Mobile sensor network control using mutual information methods and particle filters," *IEEE Trans. on Automatic Control*, vol. 55, no. 1, pp. 32–47, 2010.

[61] Z. Tang, "Information-theoretic management of mobile sensor agents," Ph.D. dissertation, The Ohio State University, Ohio, 2005.

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

[63] A. Gelman, "Objections to Bayesian statistics," *Bayesian Analyses*, vol. 3, no. 3, pp. 445 – 450, 2008.

[64] U. Zengin and A. Dogan, "Real-time target tracking for autonomous UAVs in adversarial environment: A gradient search algorithm," *IEEE Trans. on Robotics*, vol. 23, no. 2, pp. 294–307, 2007.

[65] J. P. How, K. C. K. Cameron Fraser, L. F. Bertuccelli, O. Toupet, L. Brunet, A. Bacharach, and N. Roy, "Increasing autonomy of UAVs," *IEEE Robotics and Automation Magazine*, vol. 16, no. 2, pp. 43 – 51, 2009.

[66] J. How, C. Fraser, K. Kulling, and L. Bertuccelli, "Increasing autonomy of UAVs," *IEEE Robotics and Automation Magazine*, vol. 16, no. 2, pp. 43–51, 2009.

[67] L. Lazos and R. Poovendran, "Stochastic coverage in heterogeneous sensor networks," *ACM Trans. Sen. Netw.*, vol. 2, no. 3, pp. 325–358, 2006.

[68] F. Bullo, J. Cortés, and S. Martínez, *Distributed Control of Robotic Networks*, ser. Applied Mathematics Series. Princeton University Press, 2009, electronically available at http://coordinationbook.info.

[69] B. Bollobàs, *Modern Graph Theory.* Springer-Verlag New York, 1998.

[70] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Transactions on Programming Languages and Systems*, vol. 5, no. 1, pp. 66–77, 1983.

[71] M. Khan and G. Pandurangan, "A fast distributed approximation algorithm for minimum spanning trees," *Lecture Notes in Computer Science*, vol. 4167, pp. 355–369, 2006.

[72] A. Gibbons, *Algorithmic Graph Theory.* Cambridge University Press, 1985.

[73] S. Khuller and U. Vishkin, "Biconnectivity approximations and graph carvings," in *STOC '92: Proceedings of the twenty-fourth annual ACM symposium on Theory of computing.* New York, NY, USA: ACM, 1992, pp. 759–770.

[74] R. Cortez, J. Luna, R. Fierro, and J. Wood, "Multi-vehicle testbed for decentralized environmental sensing," *IEEE International Conference on Robotics and Automation*, pp. 1052–1058, May 2010.

[75] E. M. Craparo, J. P. How, and E. Modiano, "Simultaneous placement and assignment for exploration in mobile backbone networks," *IEEE Conference on Decision and Control*, pp. 1696–1701, December 2008.

[76] E. W. Frew and J. Elston, "Target assignment for integrated search and tracking by active robot networks," *IEEE International Conference on Robotics and Automation*, pp. 2345–2359, May 2008.

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

[78] N. Michael, J. Fink, and V. Kumar, "Experimental testbed for large multi-robot teams:verification and validation," *IEEE Robotics and Automation Magazine*, vol. 15, no. 1, pp. 53–61, March 2008.

[79] M. Schwager, J. McLurkin, J. Slotine, and D. Rus, "From theory to practice: Distributed coverage control experiments with groups of robots," *Proc. of the International Symposium on Experimental Robotics*, July 2008.

[80] "Mobilerobots," Website, http://www.activrobots.com/.

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

[82] "Usarsim," Website, http://usarsim.sourceforge.net/.