COMMUNICATION AWARE MOBILE ROBOT TEAMS

James Stephan

A DISSERTATION

in

Electrical and Systems Engineering

Presented to the Faculties of the University of Pennsylvania in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2015

Dr. Alejandro Ribeiro, *Associate Professor of Electrical and Systems Engineering*
Supervisor of Dissertation

Dr. Alejandro Ribeiro, *Associate Professor of Electrical and Systems Engineering*
Graduate Group Chairperson

Dissertation Committee:
Dr. Alejandro Ribeiro, *Associate Professor of Electrical and Systems Engineering*
Dr. George J. Pappas, *Professor of Electrical and Systems Engineering*
Dr. Vijay Kumar, *Professor of Mechanical Engineering and Applied Mechanics*
Dr. Jonathan Fink, *Army Research Laboratory*
Dr. Brian Sadler, *Army Research Laboratory*

COMMUNICATION AWARE MOBILE ROBOT TEAMS

Acknowledgments

I would like to begin by thanking my advisor, Alejandro Ribeiro, for providing the proper balance of freedom to work on the problems that I found most interesting, and guidance to see them to completion. His ability to clearly articulate the specific goal we were working towards made the entire process possible. I would also like to thank the members of my committee for taking the time to provide valuable comments and suggestions throughout this process and accepting only my best.

The completion of this thesis would not have been possible without the help provided by various members of the MRSL and GRASP labs. I would specifically like to thank Ben Charrow, Philip Dames and Kartik Mohta, without whom most of the experiments in this thesis would not have been possible. I would also like to thank Jonathan Fink for the immense help and guidance in composing the results of my work into meaningful papers.

Next, I would like to thank my family for the unconditional love and support they have provided throughout my PhD.

Finally, I want to thank Arye, my best friend and love of my life. Her support and encouragement helped me get through the hardest of times. Without her, none of this would be possible.

LYSM

ABSTRACT

COMMUNICATION AWARE MOBILE ROBOT TEAMS

James Stephan

Alejandro Ribeiro

The type of scenarios that could benefit from a team of robots that are able to self configure into an ad-hoc multi-hop mobile communication network while completing a task in an unknown environment, range from search and rescue in a partially collapsed building to providing a security perimeter around a region of interest. In this thesis, we present a hybrid system that enables a team of robots to maintain a prescribed end-to-end data rate while moving through a complex unknown environment, in a distributed manner, to complete a specific task. This is achieved by a systematic decomposition of the real-time situational awareness problem into subproblems that can be efficiently solved by distributed optimization. The validity of this approach is demonstrated through multiple simulations and experiments in which the a team of robots is able to accurately map an unknown environment and then transition to complete a traditional situational awareness task.

We also present MCTP, a lightweight communication protocol that is specifically designed for use in ad-hoc multi-hop wireless networks composed of low-cost low-power transceivers. This protocol leverages the spatial diversity found in mobile robot teams as well as recently developed robust routing systems designed to minimize the variance of the end-to-end communication link.

The combination of the hybrid system and MCTP results in a system that is able to complete a task, with minimal global coordination, while providing near loss-less communication over an ad-hoc multi-hop network created by the members of the team in unknown environments.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis, we address the problem of a team of autonomous robots operating in unknown environments, while they create and maintain a communication network in a comprehensive way. This requires a unification of problems across multiple domains which include, but are not limited to, system architecture, wireless communication links, network routing, and autonomous mapping. We approach the problems in this manner because by expanding consideration to multiple domains, we have the ability to balance the competing objectives of the individual components. This enables the system to adapt to differing objectives and to optimize the overall behavior. In other words, this problem must be addressed holistically in order to enable the broadest range of scenarios in which the team can operate.

The potential applications of robot teams providing situational awareness of unexplored areas is of particular interest [16, 30, 37]. Teams of low-cost, low-power, expendable robots with the ability to explore an environment and provide real-time situational awareness without requiring a priori knowledge of the environment or existing communication infrastructure are desired. In the most general case, as the robots move through an environment, they collect measurements that must be transmitted back to a central location for exploitation. A specific application involves a first responder requiring knowledge of victim locations inside a partially collapsed building. To

accomplish this task, the robots must be able to communicate reliably over the point-to-point communication links, while efficiently moving through the environment so as to maintain the communication network, all the while operating in an unknown and dangerous environment.

Three main challenges are associated with the design and implementation of such a system. First, high-speed reliable communication over a wireless multi-hop network is difficult to achieve due to the random fluctuations in the wireless links that cause intermittent packet loss. These sporadic and difficult to predict fluctuations result in inefficient communication over the wireless channels. Second, operation in complex environments while preserving a communication network typically require high levels of global coordination. The problem with global coordination is that as robots are added to the team, the amount of ancillary coordination data on the communication network increases. This results in an upper limit for the number of robots on the team due to saturation of the communication network. Finally, in order to operate effectively, a representation of the environment is required for both motion planning and channel estimation. Unfortunately, a priori knowledge of the environment is not feasible in scenarios where robotic teams are generally the most useful.

These three challenges motivate the requirement to validate our system not only through simulation but also through experiments. Since the system is expected to provide real-time situational awareness over wireless links that fluctuate randomly, success in simulation is insufficient to claim system verification; therefore experiments are required for confirmation. Consequently, in this thesis we utilize a team of robots to empirically verify satisfaction of the system requirements, and only rely on simulations in scenarios where adequate resources are unavailable. Finally, since the focus of this thesis is on the design and implementation of the system, the exploitation of the data collected by the robots, e.g. target location or data fusion, is beyond the scope of this work.

1.1 Objective

The objective of this thesis is to develop and demonstrate a system that allows a team of autonomous micro-robots to complete a real-time situational awareness task in an unknown and harsh environment, without the requirement of explicitly considering the evolution of the communication network. To accomplish this, the team must self organize, without colliding with environmental obstacles or each other, into an ad-hoc network, over which situational awareness data can be transmitted according to network routes that adjust to the constantly changing network topology. For this system to be effective in realistic situations, the auxiliary coordination and communication of the system must be independent of the team size so as to allow for teams of arbitrary size.

With that objective in mind, the focus of this thesis is presented in three parts. The first objective is to develop and implement a lightweight confirmation transport protocol that allows for efficient delivery of data over multi-hop wireless networks. We establish the requirements and restrictions imposed on the protocol in order to allow for operation on simple low-cost transceivers that can be integrated into the robotic platform for minimal cost and effort. Experimental verification of this protocol allows us to assume in later sections that high-speed reliable communication over a wireless network is feasible using these transceivers.

The second objective of this thesis is to reduce the level of global coordination required for operation in complex environments. We begin by formulating a motivating problem statement for team task completion and a solution that achieves this while reducing global coordination by development of a novel hybrid system architecture. This architecture, which is composed of two subsystems, allows the team to operate with no global coordination after an initial period. The design of the hybrid architecture is concluded with experimental results both verifying successful task completion, while preserving the communication network and comparing its performance to other such systems.

The final objective of this thesis is to allow for operation in unknown environments. We begin by formulating the problem of autonomous exploration and examine the consequences of adding communication requirements. This leads to a solution that requires the hybrid system be augmented to allow for operation in dynamic environments while still preserving the minimal global coordination. Our consideration of this problem concludes with experimental verification that our system allows a team of robots to operate in unknown environments while providing real-time situational awareness.

1.2 Literature Review

This thesis, as with most research in robotic teams, spans a wide variety of separate research areas. These topics range from communication over ad-hoc wireless networks, to communication aware motion control, to network optimization, to autonomous mapping. In this section we will review the relevant literature beginning with an investigation of various methods used to communicate reliably over ad-hoc networks. This is followed by an examination of systems that provide communication-aware motion control, i.e. the ability of a team of robots to move through an environment while maintaining certain communication constraints. Some constraints explored are simple connectivity, minimum number of hops between source and destination, and end-to-end data rate. Next, with an understanding of the systems that maintain the underlying communication network, we examine recent advancements in optimal packet routing over ad-hoc wireless networks. We conclude with a brief study on autonomous mapping and exploration, where we examine recent work in information theoretic exploration with a focus on optimal sensing locations.

1.2.1 Communication Protocols Over Wireless Links

Communication protocols have existed since the days of the first packet switching networks of the 1950's. The main responsibility of a communication protocol is to manage the transport layer of the OSI stack, [99]. The OSI stack is a standardization that allows designers to abstract the different layers necessary for a computer network to operate. The transport layer, the space where communication protocols exist, is above the network, data link, and physical layers, but below the session, presentation, and application layers. With this location in the stack, communication protocols have the responsibility of providing end-to-end communication services to the upper layers, while providing various low level capabilities, such as flow control and reliability.

The most well known communication protocol is the Transmission Control Protocol (TCP) [11], which was originally proposed in 1974 by Cerf and Kahn. This protocol was designed to optimally control the flow of packets from the source to the destination, by means of a connection oriented link. One benefit of their approach was a guarantee that packets sent by the source would be received by the destination in the order they were transmitted. The is achieved by the receiver sending an acknowledgment packet, or ACK, to the sender when a packet is received. To further improve link utilization, TCP combines multiple packets into a segment which is transmitted as a logical block. The size of the segment is determined by a congestion control algorithm which implements a sliding window, [80]. The term sliding window is used because as segments are confirmed, by reception of an ACK for every packet in the segment, the size of the window, and subsequent segments, increases. This process of expanding the window increases until the link is saturated and packets begin dropping, mainly due to congestion with other users of the link. To detect when a packet is dropped, the sender uses a timeout based on an estimate of the round-trip time, which is the time necessary for a packet to go from the sender to the receiver and back. If a packet has not been confirmed within the allotted timeout, a retransmission is initiated and the window is contracted, reducing the size of subsequent segments.

While the ability to reliably transmit data in order over a network was a major breakthrough, there were still applications in which the timeliness of the data was most important or the overhead associated with TCP was too burdensome. This lead to the development of the Universal Datagram Protocol (UDP) [62] in 1980. In contrast to TCP, UDP provides no guarantees on the reception or order in which packets arrive at their destination, but by forgoing the overhead associated with TCP, UDP is able to achieve higher data rates.

These protocols and their offshoots were meticulously tuned and optimized for use in wired networks, which until recently were the only option. The transition from wired to wireless networks presents a paradigm shift in the assumptions and properties of the communication links. Specifically, the medium in which the data is transmitted is no longer a physical substrate but instead radio waves. The transition from a physical link, such as copper or fiber optics, to radio waves introduces a large variability in the signal strength at the receiver, which is known to relate directly to the capacity of the channel, [15]. In a wired network the signal strength does not fluctuate due to the consistent propagation of the data over the physical link. This is not true for wireless links, which exhibit random fluctuations in the propagation of the radio wave. To quantify the effects on propagation, three levels of modeling are employed, called scales.

The first and largest scale is related to the power lost as the wave propagates away from the sender, which is purely a function of the distance the wave has to travel. The second, or middle scale, incorporates objects in the environment and describes how they affect radio wave propagation, called shadowing, because the object effectively casts a shadow on the receiver with respect to the sender. These two contributions have been modeled deterministically since the effects are relatively stable over time and relate to the dominant path, or the direct line between the sender and receiver. The third and smallest scale considers fading, which is the reflection and refraction of the radio wave off objects in the environment. This process results in the same signal traveling a variety of paths before reaching the receiver. As the signal travels along these multiple paths, it reaches the receiver at different times, resulting in interference, which degrades

performance. This level of interference, and subsequently signal strength, has been shown to vary dramatically due to small movements by the sender, receiver, or object in the environment, [64].

It is these random fluctuations in the channel capacity, first noticed by Caceres and Iftode in [8], that cause degradation in the performance of TCP over wireless links. The effects of random fading can most evidently be seen in the random loss of packets as they travel over the channel and the resulting sub-optimality of TCP. This behavior is also explored in [91] where Xylomenos and Polyzos experimentally compare the performance of UDP and TCP over wireless links. Their experiments reveal lower then expected performance over the wireless links, due mainly to packet collision and poor recovery from dropped packets. The reason for the increase in packet collisions is that unlike a wired network where the point-to-point links are isolated from each other because they operate over physically separated wires, the links in a wireless network are not isolated. The communication between one pair of nodes can be interfered with by the communication between two other unrelated nodes, as well as by other radio waves in the area. Since this is a physical phenomenon, best addressed at the physical layer of the OSI model, most communication protocol research do not actively consider this. In contrast to packet collisions, random packet loss and methods to mitigate their affects on TCP is an active research area.

From the beginning of wireless networks and the identification that traditional TCP is inadequate, there have been numerous modification and suggested improvements for its use over wireless links. One set of prosed modifications focus on selective retransmission schemes. These approaches allow the sender to identify the specific dropped packets and to only retransmit those before performing a window contraction. In [38] Keshav and Morgan propose a modification, called Simple Method to Aid Retransmit (SMART) which decouples congestion control from flow control by including in the ACK packet both the cumulative ACK count and the sequence number of the pack that initiated the ACK. This allows the sender to identify the packets that are lost and allows for selective retransmission. This process was later modified and included in the TCP standard [25], named Selective Acknowledgment (SACK), where the options field of the TCP header

is used to identify to the sender the packets in the segment that were not received. This allows the sender to identify the exact packets that were dropped and limit retransmission to only those packets, reducing used bandwidth and preventing overly aggressive window contraction. With its introduction in the TCP standard, SACK is commonly used to reduce extraneous retransmissions.

Another method for identifying when a packet is lost due to the random fluctuations of the channel is proposed by Tsaoussidis and Badr in [85]. In this protocol, when confirmation is delayed the sender enters into a probe cycle as opposed to contracting the window. In the probe cycle, small probe packets are sent to the receiver, which responds with an ACK if received. Upon confirmation of two successive probe packets being received, the sender returns to normal operation without contracting the window. If the errors persist, the cycle exits and the window is contracted, since this is an indication that the channel quality is reduced and the dropped packet was not due to random fluctuations.

A major advancement in TCP over wireless networks occured with the introduction of TCP Santa Cruz [58] by Parsa and Garcia-Luna-Aceves. In this modification, the window based congestion control process was enhanced to use the relative delay between packets in the forward direction, which provides robustness to ACK loss and reverse path congestion. By leveraging the relative delay between two packets, the system is able to identify when loss is due to channel congestion and when it is due to random loss. Additionally, this modification provides a better estimate of the round trip time, resulting in a reduction of the number of times the window is contracted inappropriately.

The use of information other than ACKs to augment the congestion algorithm is also explored by Casetti et al. in [9], where they propose TCP-Westwood. This modification uses a metric for effective bandwidth that is computed by the rate of returning ACKs. This value is used to control the reduction in the size of the window. This approach allows for the window to be contracted in accordance with the effective bandwidth, as opposed to the common halving used in traditional TCP. TCP-Westwood was demonstrated experimentally to provide greater than five times the

throughput of traditional TCP.

The use of a window for congestion control is not the only method explored in the literature. In [73], Sinha et al. utilizes a rate-based transmission control scheme in lieu of the self-clocking window scheme. In this scheme, ancillary data about the average delay between packets for both the sender and receiver are used to adjust the rate at which packets are sent. While in [1], Aggarwal et al. propose ACK Pacing that attempts to ameliorate the bursty nature of communications by performing rate control on the ACK transmission from the receiver. By smoothing out the rate at which ACKs are sent, the likelihood of packet loss in the reverse direction is lowered, at the expense of throughput.

In work done by Fu e al [26], they identify an optimal window size for TCP given a specific network topology, but existing window control methods rarely operate with that window size. To that end, they propose an adaptive pacing system that seeks to determine the optimal window size, which results in up to a 30% performance improvement over TCP in simulation. Building upon the concept of adaptive pacing in [19] ElRakabawy et al. proposes a system, TCP-Adaptive Pacing (TCP-AP), that is a hybrid between rate based control and congestion window methods. These modification are examined in simulation and demonstrate significant gains over existing methods.

There have been multiple publications that survey and compare many of the proposed modifications in TCP. One of the first was done by Pentikousis et al. [59] in which they survey and evaluate a variety of TCP modifications for use in networks that contain a mix of wired and wireless connections. They conclude the poor performance of traditional TCP to a number of factors, including user mobility and random losses. A more recent survey done by Al-Jubari et al. [3] provide an extensive review of these and other protocols with the conclusion that the proposed methods, while effective, are specially designed to meet the needs of the application and environment and thus are not easily generalized.

1.2.2 Communication-Aware Motion Control

Even if reliable communication over wireless links can be guaranteed, the motion controllers on the robots must be aware of the affect motion has on the communication links. Therefore, the ability for a team of mobile robots to move through an environment, while maintaining specific properties of the underlying communication network is of particular interest to this thesis. This ability is complicated by a variety of factors such as node mobility, wireless channel estimation, dynamic network topologies, and non-convex environments. These complications are inherent to the problem of communication-aware control and therefore must be understood before further progress can be made.

The use of communication over wireless links is well established in the literature. In an early publication by Jadbabaie et al. [33] they provide a theoretical understanding of the flocking example given by Vicsek et al. in [86]. In this paper, they use graph theory and dynamical system theory to provide a theoretical explanation and convergence results for the observed behavior that a collection of n autonomous agents can form into a moving flock by using only local measurements. In [65], Ren and Beard continue the work done by Jadbabaie et al. and demonstrate that consensus can be reached by either a discrete or continuous update scheme, in an asymptotic sense over dynamically changing networks as long as the union of directed interaction graph has a spanning tree frequently enough. These two works influenced Olfati-Saber et al. in their work on extending the problem formulation to applications such as coupled oscillators, formation control, fast consensus in smallworld networks, and gossip algorithms, [56]. They also provide simulation results that demonstrate the effect smallworld networks have on the speed of consensus algorithms, and cooperative control.

Leveraging the previous work on consensus algorithms over dynamic network topologies, in [4] Antonelli et al. provide a distributed system that is capable of driving a team of robots through an environment while maintaining a formation. This is achieved by utilizing a distributed controller-

observer scheme that is able to accurately estimate and control the time varying-centroid of the formation. This is demonstrated in experiments involving multiple wheeled robots.

While these systems are able to operate over dynamic network topologies they do not consider the affect motion has on the communication links, which change as the robots move, and how those changes affect the network topology. This highlights the need to consider how mobility and communication interact. In the work done by Hsieh et al. [31,32] they demonstrate that a team of robots can independently move through an environment using the gradient of the received signal strength as an indicator of the communication network. They show through experiments that end-to-end connectivity can be maintained by a reactive controller in an urban environment. This method eschews a direct modeling of the point-to-point communication link and is still able to provide acceptable results.

While the work by Hsieh et al. is promising, there has been substantial research dedicated to mobility and communication using a variety of channel estimation techniques. The first, and most widely researched method, due to its simplicity, is a geometric disc model, where communication is assumed if two robots are with fixed communication range. This leads to a binary representation of the network and the ability to represent the communication requirements as a set of geometric constraints. In [74] Spanos and Murray use a metric they call connectivity robustness to quantify how robust the network topology is to movement by the individual robots. They demonstrate that imposing constraints on the motion in this form allows for a distributed motion planning algorithm to be used so as to allow more freedom of motion and ultimately a larger reachable set of formations, when compared to the use of direct connectivity constraints.

Similarly, in [55] Notarstefano et al. consider robots with double integrator dynamics and how this affects communication-aware motion. In this work they demonstrate that it is feasible for a team to maintain connectedness by utilizing what they refer to as a double-integrator disk graph, and that given desired control inputs produces the closest match that respects the connectivity constraint. This can be implemented via distributed "flow-control" algorithm. In [12], Chakraborty

and Sycara propose a distributed control system that solves a convex optimization problem that seeks to maintain k-connectedness. This allows the robots to independently determine their motion towards a desired location while preserving the network connectivity constraints, using only local information via an incremental algorithm. They conclude from the simulation results that the runtime for the incremental algorithm is independent of the team size. Thus, the process can scale efficiently.

The next method used in communication aware motion control employs a smooth decaying function of the inter-robot distance. This method is useful because it is able to capture the large scale path loss and possibly shadowing when estimating the channel. The consideration of non-binary links opens the field to leverage many of the advancements from graph theory, specifically algebraic connectivity or the second smallest eigenvalue of the Laplacian, which measures the rate of information spread over the network. The work done by Kim and Mesbahi considers weighted graphs, where the vertices are able to move subject to some proximity constraints and in which the edge weight between two vertices is a function of their separating distance [39]. They attempt to determine the optimal placement of the vertices so that the second smallest eigenvalue of the Laplacian is maximized. This work, while not specifically considering robots, can clearly be translated to robotic teams.

Considering robotic teams explicitly, De Gennaro and Jadbabaie propose a potential based control law which follows a supergradient to increase the second smallest eigenvalue of the Laplacian [18]. This system operates in a distributed manner and models the link between robots as an exponential decay. This work was built upon by Ji and Egerstedt in [34] to impose the requirement that the team also complete some objective. The two objectives examined in this publication are rendezvous and formation control. Zavlanos and Pappas [95] continue this line of research by translating the connectivity constraints into differential constraints on each robot's motion by taking the dynamics and spectral properties of the Laplacian into consideration . This is implemented as artificial potential functions that drive the agents away from formations that

produce undesirable results, with respect to connectivity, while avoiding collisions with each other. In a different branch of the work done in [18], Stump et al. [81] present a system that is able to provide a minimum level of connectivity to an exploring robot back to a stationary access point in a walled environment. This is achieved by designing a controller that moves in the direction that provides a maximum increase in the second eigenvalue of the Laplacian while keeping the number of robots between the explorer and the access points below a maximum value. This approach is shown to provide acceptable communication quality even when the hop-count requirement is violated.

In contrast to the myopic actions prescribed by the previous systems, Schuresko and Cortes propose a distributed approach based on game theory in which the robots determine if a collective action will violate the connectivity requirement [70]. The connectivity measure considered is still the second smallest eigenvalue, but by taking this approach their algorithm can operate even with imperfect information caused by communication delays or robot motion. The flexibility afforded by this approach allows the algorithm to operate in non-convex environments, i.e. those that contain obstacles.

In [49, 96] Zavlanos et al. present a distributed motion algorithm based on key control decomposition that uses the discrete space of graphs to control the structure of the network. Local estimates of network topology and algebraic connectivity are used to identify links that can be deleted, with ties settled by means of a gossip algorithm. In parallel the motion controller operates to maintain existing links through nearest-neighbor potential fields. This system is demonstrated in simulation and experimentation with non-holonomic robots.

In a shift away from algebraic connectivity, Tekdas et al. examine a problem similar to Stump et al, in that there is a mobile user that wishes to maintain connectivity with a stationary access point [83]. To accomplish this task Tekdas et al. leverage advanced motion planning algorithms based on models of the users motion that maximize the amount of time until the user's motion cause a break in the link. The system's ability to maintain connectivity is demonstrated in

experiments when the path of the user is know a priori and when the user is adversarial.

The shift away from algebraic connectivity is continued in [97,98] where Zavlanos et al. change the interpretation of network integrity to the end-to-end data rate, not the algebraic connectivity of the communication graph. This change in the interpretation requires that the system compute the optimal packet routes for the current formation. This is achieved by solving a distributed optimization problem in which neighboring robots share pertinent information. For motion control, a continuous time controller utilizes a navigation function approach that includes virtual communication obstacles. This system is shown in simulation to provide the desired end-to-end rate while completing a secondary objective.

The third class of methods used to model the point-to-point channels incorporate the effects of multi-path fading. This is done by interpreting the channel quality as a random variable with the mean determined by the expected path loss and shadowing, while the variance is determined by the fading characteristics of the environment. In the work done by Mostofi et al, they provide a framework for accurately characterizing the probabilistic nature of the channel, by taking into account the effects of all three scales of signal loss [52]. This work provides a comprehensive overview of channel modeling, expressly for use in mobile robot teams, which is subsequently verified by experimentation.

Leveraging this framework, Mostofi presents motion planning strategies that are able to maintain communication, by learning the parameters for the environment through measurements such as signal-to-noise ratio and correlation characteristics [51]. The main contribution of this work is a randomized motion planning strategy that allows robots to escape deep fades, either correlated or uncorrelated, which can greatly degrade system performance.

Another system that incorporates a probabilistic model of the channels can be found in [21, 23,24]. In this work Fink et al. proposed a system that provides a minimum end-to-end data rate, similar to the metric in [97,98], while the team moves through a complex environment to complete a task. In this system, the optimal routes for the packets are determined by an optimization

problem where the minimum end-to-end rate is satisfied above a specified probability threshold. A randomized motion planning strategy is employed to determine the trajectories of each robot in the joint configuration space, such that the probability threshold is satisfied at all times.

The problem of optimal placement of mobile agents to construct a relay network between two fixed locations is examined in [93] by Yan and Mostofi. In this problem, they consider minimization of the bit error rate of the end-to-end channel as opposed to the data rate or second smallest eigenvalue of the Laplacian. Using a probabilistic model for the channel and a robotic router motion optimization they are able to construct a distributed system that is able to learn the environmental characteristics and operate in the presence of obstacles. They also demonstrate that minimization of the bit error rate as opposed to maximizing the second smallest eigenvalue of the Laplacian results in better performance in complex environments.

The purpose of these methods is to provide a model for the communication link that allows for efficient motion planning, but even the methods that provide online estimation of the channel are restricted to a generalization of the environment and require collection over large areas. To overcome this, Gil et al. introduce a novel system for channel estimation and optimal motion control that relies on methods developed for RADARs, specifically Synthetic Aperture Radar (SAR) [27]. They demonstrate the creation of a SAR system using commercially available hardware and software to accurate predict the direction of motion that increases the signal to noise ratio the most. This is made possible by coherently integrating the received signal as the robot follow a straight line, effectively creating a large receive antenna, which can be used to determine directionality of the signal.

1.2.3 Network Optimization

As the ability of teams of robots to move through complex environments increases we must also consider the problem of wireless network optimization. As previously discussed, there has been considerable research in optimal transport protocols over wireless links, but as the robots move

and the network topology changes, the lower layers of the OSI model, specifically the routing layer, must also adapt.

Some of the first solutions to optimal routing over networks with dynamic topology focused on minimization of the number of hops a packet must take to reach its final destination. These protocols, namely Dynamic Source Routing proposed by Johnson and Maltz [35] and Ad-Hoc On-Demand Distance Vector routing proposed by Perkins et al. in [60], rely on real-time measurements of the channels to infer the network topology and thus determine the routes that result in the minimum number of hops a packet must take. These systems result in sub-optimal solutions when applied to realistic ad-hoc networks, due mostly to the inaccuracy of determining the existence of a link. As demonstrated in [47] by Lundgren et al, there are regions where a probe signal, called the HELLO, can be successfully received over a link but data cannot. This results in a gray-zone where the link is believed to exist but data transmission is impossible.

In the work of De Couto et al, they propose a different metric for optimal routing, called expected transmission count [17]. This metric does not assume that the path with the least hops is optimal; and instead, they characterize the paths by the expected throughput, which they maximize. This approach leads to routing solutions where the optimal route may include more hops but results in significantly higher data throughput, often by a factor of two over the approaches that minimize hop-count.

Other proposed solutions for network optimization rely on a cross layer approach that controls multiple layers of the OSI model simultaneously. While this does violate the abstraction imposed by the OSI model, the interaction between the lower levels of the model in wireless networks is much greater than in traditional wired networks. To that end, one solution proposed by Eryilmaz and Srikant in [20] solves the problem by jointly scheduling the routing and congestion control, in order to provide asymptotic guarantees on the stability of the buffers and the fairness of network resource allocation. This is achieved by sharing the queue lengths across the layers of the network.

In another solution which controls the transport, network, and physical layers in unison, Lin

et al. propose a "loosely coupled" cross-layer solution [45]. This solution allows for optimization of the separate layers with minimal inter-layer coordination, which aside from the physical layer can be solved optimally in a distributed manner.

In the spirit of returning the independence of the OSI model, Yi and Shakkottai present a solution that provides congestion control for a multi-hop wireless network by formulating an optimization problem, where one of the constraints is a channel access time constraint to symbolize the time-division strategy used in the lower layers, [94]. They demonstrate that in the absence of delay, this solution provides globally stable results, and in the presence of delay, areas of high load spatially are spread out over the network, resulting in bounds on the peak load of a node.

Just as the trend in communication-aware motion control moved towards probabilistic models of the channel, so to did the field of network optimization. In the work done by Ribeiro et al. in [66], they consider the inter-node links to be random quantities with know mean and variance. This approach lends itself to solutions where the decision of which node a packet is transmitted to next is determined by a probability distribution. They show that this probability distribution can be determined by a convex optimization problem which can be efficiently solved via interior point methods.

Continuing with this approach, Wu et al. proposes an extension where solutions are determined by either maximizing an average utility subject to variance constraints, or minimization of variance subject to minimum average utility [89]. They continue by showing that both of these problems can be formulated as convex optimization problems that can be solved in a distributed manner, due to the separability of the problem. They conclude with a comparison of the resulting solutions to those found by a centralized system showing no performance penalty even with a significant reduction in the overall communication.

This work is further refined by Ribeiro et al. in [67], in which they demonstrate that rate-oriented criteria such as minimum rate, weighted sum of rates, product of rates, and sources rate can be maximized by means of a stochastic routing solution. This solution is obtained by

a distributed algorithm in which dual variable are exchanged and optimality can be guaranteed under mild conditions.

1.2.4 Autonomous Exploration

The ability of a robot to move safely through an environment and perform a task is predicated on successful localization. This means that given information from an onboard sensor such as a laser range finder or camera the robot can estimate their location in the environment with high accuracy. There are configurations in which external sensors, such as a motion capture system, are used to provide a highly accurate estimate of the robot location, but due to their cost and the effort to calibrate, they are relegated to use in laboratories and fixed installations. Since this thesis is focused on operating in realistic situations, the use of these systems is not considered.

In order to successfully localize, the robot first needs to have a representation of the environment. The representation of the environment used to localize is typically dictated by the types of sensory inputs available to the robots. For example, if the robot is equipped with a laser range finder that can accurately measure distances along multiple angles simultaneously, then it is common to use an occupancy grid representation of the world, [84]. An occupancy grid is a representation of the world in which areas or volumes, depending on whether the map is 2-dimensional or 3-dimensional, are discretized into cells that represent that specific location in the environment. Another common representation, used when the available sensor is a camera, is a landmark based map, [6]. In a landmark based map, not every single location is represented, instead only important and easily identifiable features are represented. This allows for a much more compact representation of the world compared to an occupancy grid, but also introduces a higher probability of decreased accuracy due to insufficient features.

While the ability to localize in a given map is useful, it is limited to areas that have already been mapped. Since the world is constantly changing and there are always new places to explore, this problem has been extended to consider Simultaneous Localization and Mapping (SLAM). One

of the first explorations of the problem was done by Leonard et al. in [43], where they presented the problem as the "which came first, the chicken of the egg?", since the robot cannot localize without a map and it cannot create a map without knowledge of its location. In order to solve this problem, they proposed a system that jointly determines the location of the robot and maps the surroundings, via onboard sensor measurements. As more measurements are obtained, the accuracy of the estimated location and map are increased and refined.

This process was later extended by Castellanos et al. in [10] where they introduce a framework that uses a probabilistic representation of uncertain geometric information. This allowed for much higher accuracy of the robot's pose and thus more accurate maps when using a 2-dimensional laser range finder. The advancements in SLAM have not been limited to robots equipped with laser range finders; in [71] Se et al. propose a SLAM system in which the main sensory input was a stereo vision system. They demonstrated that using this input they were able to successfully move through a realistic environment while mapping naturally occurring features in the environment. The ability to perform SLAM in larger and more complex environments is the topic covered by Olson, [57], in which he presents an optimization algorithm to efficiently estimate the most probable representation of the environment given previous noisy measurements.

In parallel to advancements in SLAM algorithms, the problem of autonomous exploration was evolving. Autonomous exploration is the process by which the robot, not an operator, determines the optimal actions given an objective. The objective could be to locate a target in the environment, such as a person or object, or more simply, the robot could be tasked with creating an accurate and complete map of the environment. In that regard, some of the earliest work done by Yamauchi [92] focused on the boundary between known and unknown spaces. These boundaries are called frontiers since they are on the frontier of the known space. In this work, Yamauchi demonstrates that driving the robot to frontiers ultimately results in a complete map of the environment regardless of its complexity. Since then there have been countless publications on the optimal strategy for selecting the "best" frontier. One such paper [29], Holz et al. provides

a comparison of various techniques for frontier selection and comes to the conclusion that the optimal strategy is to move to the closest frontier.

Recent advancements in the selection of optimal robot motion in autonomous exploration has focused on using an information theoretic approach. These approaches use various information theory metrics to decide the optimal location of the next set of measurements. In order to use information theory, these approaches take a probabilistic approach and represent the maps of the environment as an occupancy grid, where a cell contains the probability of an obstacle residing at that location. The work done by Kollar and Roy [40] includes Shannon's entropy [15] into the motion control of the robot. In this work, they reduce the map to a skeleton graph and determine the minimum entropy path through the environment by means of an optimization problem that uses gradient ascent. The resulting trajectories offer improved accuracy of the resulting map. Similarly, the work done by Stachniss et al. [75] also includes Shannon's entropy in the objective and select actions that maximally reduce the entropy. Another metric used is the a-optimal information measure, which is used by Sim and Roy in [72]. In this work, they present a method for determining non-greedy global planning trajectories that result in accurate maps by attempting to close loops, while restricting the planning to an appropriate control policy class.

In [7] Bourgaul et al. they use Shannon's mutual information [15] in their objective function to expedite the mapping process. Hoffmann and Tomlin improve on this work in [28] where they provide an efficient method from computing the mutual information when using a particle filter. Even with this efficiency gain, as the environments become larger and the sensor measurement history grows, the computational tractability of these systems comes into question. As shown in [36], Julian et al. highlight the limitations of Shannon's mutual information due to the requirement to numerically integrate over all possible maps and all possible robot locations. This is the problem studied by Charrow et al. in [13] where they present an approximation of Shannon's mutual information. They demonstrate that this approximation allows for real-time operation through extensive experimentation.

Building upon their past work, Charrow et al. [14] then considers another information measure, specifically Cauchy-Schwarz Mutual Information (CSQMI) [61]. While not identical to Shannon's mutual information, CSQMI behaves similarly to Shannon's Mutual information and has the added benefit of computational tractability. This allows Charrow et al. to design a real-time exploration system that is able to achieve the same level of accuracy as the systems that rely on Shannon's mutual information, for only a fraction of the computation cost.

1.3 Approach

The objective of this thesis is to present and experimentally verify a system that can allow a team of robots the ability to provide real-time situational awareness by means of a self configured ad-hoc wireless network without a priori knowledge of the environment. This problem can be broken down into three parts: first, the development of a lightweight confirmation transport protocol; second, the design of a hybrid system architecture that reduces global coordination; third, the integration of simultaneous localization and mapping.

In the first part, we begin by examining the common transport protocols that are used in multi-hop wireless networks - TCP and UDP. The design of the protocol is driven by the desire to use low-cost simple transceivers to create a reliable wireless network. The decision to limit the capabilities of the transceivers requires that the protocol be as lightweight as possible while still providing a high quality of service. We continue by detailing the evolution of the protocol from simple acknowledgment to an optimized protocol designed with these transceivers and wireless ad-hoc network in mind. We demonstrate the benefit of using the protocol during experiments in which a team of robots move through an environment while maintaining a communication network.

In the second piece, we begin by considering the completion of an arbitrary task by the team. We begin with an examination of the benefits and drawbacks of prior systems - centralized and

distributed. Next we propose a system architecture that completes the task with minimal global coordination. The architecture take a hierarchical approach to the problem by initially solving for a global solution and then using that as a guide for dynamic motion and network control during system operation. To generate the initial solution, we approach that problem of concurrent mobility and communication to determine feasible trajectories for each robot that preserve the network, which is closely related to [24].

In order to determine the motion and network routing during operation, we solve a modified version of the global problem which can be solved in real-time via a distributed algorithm. This method is closely related to [98]. By using the global solution as a road map, this system is able to decompose the larger problem into subproblems that do not require global coordination. The real-time operation also allows the system to dynamically adjust to changes in the environment. We demonstrate though experimentation that this approach results in improved performance over an existing system while reducing the amount of global coordination.

In the final piece, we remove the requirement of a priori knowledge of the environment. By removing a priori knowledge, we must modify the problem formulation so that we only rely on collected information. This leads to an inversion in the system architecture which increases the importance of the distributed portion of the system. Additionally, the system must now construct a meaningful representation of the environment, which requires the integration of an autonomous mapping component [14]. Finally, since the objective is real-time situational awareness, we demonstrate our system's ability to efficiently map the environment and then transition to another task, all while supporting real-time data transfer.

Since this thesis relies on constructing a robust system that can operate in unknown environments, we present experimental validation of the system where ever possible. In each set of experiments multiple locations are used to demonstrate robustness and those same locations are revisited in subsequent sections. This allows for a performance comparison as constraints are placed on the system.

1.4 Thesis Contribution

This thesis makes several novel contributions to the situational awareness problems faced by teams of robots.

One contribution is the design, development, and implementation of MCTP, a lightweight transport protocol specifically designed for use over ad-hoc multi-hop wireless network consisting of low-power low-cost transceivers. This protocol is designed to leverage the spatial diversity found in mobile robot networks and robust routing solutions. The lightweight design allows for operation over transceivers found on micro-autonomous robots while still providing reliable communication. The efficient design of the confirmation mechanism allows the protocol to operate effectively even when the number of robots on the team increases dramatically.

Another key contribution is the decomposition of the real-time situational awareness problem. By deconstructing the problem we are able to identify that the system design space can be interpreted as a 2-dimensional, as opposed to a 1-dimensional, space. This enables us to discuss systems not just as centralized or distributed, but instead consider the required level of coordination along with the optimality of the solution.

This leads to a hybrid systems approach to the problem that results in a system that is able to provide real-time situational awareness with minimal global coordination. We achieve this by leveraging highly capable local controllers to complete a task while preserving network integrity. The system is able to operate in an almost completely distributed manner, requiring minimal global coordination. Utilizing this approach, a team of arbitrary size is able to successfully navigate complex environments while providing real-time situational awareness to an operator located at an access point.

Another contribution of this thesis is an extension of the real-time situational awareness problem to unknown environments and a system to satisfy the requirements. We achieve this by augmenting the components of the hybrid system that require information about the environ-

ment, so that they can operate with incomplete information. With the introduction of a dynamic environment the reliance on the local controllers is increased, and global coordination is only used when the team is trapped in a local minima.

The final contribution of this thesis is the integration of an Information Theoretic Exploration system into the hybrid system. This results in a system that is able to autonomously map an environment while simultaneously satisfying the real-time situational awareness task. When the environment is sufficiently mapped, this final system is able to transition to a traditional situational awareness task.

1.5 Thesis Outline

The remainder of this thesis continues as follows

Chapter 2 begins by presenting a generic problem formulation of the concurrent mobility and communication problem. This formulation will serve as the basis for real-time situational awareness problem. Then we examine the way in which the network integrity constraints can be formulated, either robust or non-robust. Next, we provide a taxonomy of the system design space for adequately solving the concurrent problem, in which system design is interpreted as a 2-dimensional problem. We finish with a description of the experimental platform, software framework, and common experimental environments used in this thesis.

Chapter 3 details the creation of MCTP, a lightweight transport protocol that is specially designed for mobile ad-hoc wireless networks. We begin with a set of experiments that demonstrate the increased reliability that robust routing solutions provide over non-robust solutions in mobile ad-hoc robot networks. Next, we examine the two most common protocols for wireless communication network, resulting in a understanding of the deficiencies and benefits of each. Then we detail the construction of a transport protocol that is designed with our simple transceivers and multi-hop robust routing in mind. This transport protocol utilizes multiple point-to-point packet

receive confirmations as opposed to end-to-end confirmations used in traditional protocols. This protocol is then integrated into an existing communication-aware system to demonstrate the ability to provide near loss-less communication over a lossy ad-hoc network. The chapter concludes with a demonstration of the protocol design as the number of robots on the team increases, to the point where traditional protocols are infeasible.

Chapter 4 presents our hybrid systems approach that is fundamental to this thesis. We begin with an overview of the system architecture, composed of a two-stage feedback loop where an outer loop is responsible for infrequent global coordination and the inner loop is responsible for motion and network routing at the robot level. We then detail the construction of a global planner that is capable of determining trajectories for each robot through complex environments. This planner is designed to produce solutions that are robust to channel misestimation and trajectory deviation. We then develop a distributed controller that is able to use and improve upon those trajectories while avoiding local minima. The chapter then provides a detailed examination of the system in both simulation and experimentation, in which the benefits of the hybrid approach, specifically dynamic motion control, trajectory optimization, and operation with large teams, are demonstrated. In the conclusion of this chapter we demonstrate the system's ability to provide real-time situational awareness in the realistic scenario of patrolling a series of hallways, with and without the protocol developed in Chapter 3.

Chapter 5 integrates the approach taken in Chapter 4 and applies it to operation in unknown environments in order to produce a Simultaneous Communication-Aware Localization and Mapping (SCLAM) system. We begin by examining the effects that removing a priori knowledge of the environment has on the system designed in Chapter 4. This requires that the channel estimation and motion planning consider the uncertainty in the environment when operating. We then integrate recent advancements in efficient autonomous mapping to provide knowledge of the environment. The resulting map updates induce a dynamic communication requirement that the team must adjust to in order to provide minimal information delay. Next, we demonstrate the

system operating in multiple unknown environments, through both simulation and experiments. The chapter concludes with a final demonstration of the system, providing real-time situational awareness while initially mapping an unknown environment, then transitioning to support a highly agile aerial platform that patrols the newly mapped area.

Chapter 6 finishes with some concluding remarks and thoughts on future work.

Chapter 2

Background & System Configuration

We are interested in systematically addressing the components necessary for a team of robots to provide real-time situational awareness in unknown environments. These components include reliable communication over the ad-hoc multi-hop network created by the robots, concurrent mobility and routing with minimal global coordination, and online construction of a representation of the environment. In this chapter, we begin by formulating the generic problem of a team of robots completing an arbitrary task in a complex environment while maintaining a communication network. This problem formulation will help inform the decisions made in regard to each component of the system. We continue by examining the different approaches taken to solve this problem. Finally, we conclude with a description of the robotic platform, software architecture, and primary environments used for simulations and experimentation in this thesis.

2.1 Situational Awareness Problem

Situational awareness can be defined in a variety of ways depending on the specific scenario. For some scenarios, the term could mean searching a building for possible survivors, following a designated first responder through an environment, or providing a surveillance perimeter around a region of interest. All of these scenarios require a team of robots to successfully navigate complex environments to complete an arbitrary task while obtaining measurements from sensors, such as cameras and microphones. While some of these situations do not require real-time transmission back to an access point, they all can benefit from such a property.

Since the existence of a wireless network over which these measurements can be transmitted is highly unlikely, the robots themselves must construct and maintain a communication network. This motivates our system design to allow for real-time transmission of sensor data over an ad-hoc wireless network while the team completes a given task.

We begin by considering a team of $N-1$ mobile robots operating an a known environment and a single access point collocated with a human operator. The location of robot i at time t in the environment is $x_i(t) \in \mathbb{R}^2$, and the formation $\mathbf{x}(t) \in \mathbb{R}^{2N}$ is the aggregation of the $N-1$ robot locations and the location of the access point. Without loss of generality we assign the index $i = N$ to the access point. At time t_0 the team is deployed into a formation, $\mathbf{x}(t_0)$. The team is then given a sensing task $\Gamma(\mathbf{x}(t))$ that must be accomplished by time t_f. The sensing task Γ can be any scalar convex function that maps formations to a real value, $\Gamma : \mathbb{R}^{2N} \to \mathbb{R}$, with a minimum, Γ_{min}. When the trajectory of the team results in the minimum, $\Gamma(\mathbf{x}(t_f)) = \Gamma_{min}$ we say that the task has been completed. For example,

$$\Gamma(\mathbf{x}(t)) = \|x_1(t) - x_g\|^2, \tag{2.1.1}$$

can be used to signify that any formation in which robot $i = 1$ is at location x_g at time t_f

completes the task. We model the kinematics of a single robot as a single input control system

$$\dot{x}_i(t) = f(x_i(t), u_i(t)), \tag{2.1.2}$$

where $u_i(t)$ is the input. We assume that the robots are fully controllable, which allows us to consider a simpler motion model of the form

$$\dot{x}_i(t) = u_i(t). \tag{2.1.3}$$

This allows us to model the evolution of the formation as an integral of the control inputs from time t_0 to time t,

$$\mathbf{x}(t) = \int_{t_0}^{t} \dot{\mathbf{x}}(s)\ ds + \mathbf{x}(t_0). \tag{2.1.4}$$

Therefore, our goal is to determine control inputs $\dot{\mathbf{x}}(t)$ such that the team is able to successfully complete the task while avoiding collisions with both environmental obstacles and other members of the team. To that end, we define the set of all the formations in which a robot collides with an environmental obstacles, $\mathcal{F}_{ro}$. We then define the set of all formations that result in robots colliding with each other, $\mathcal{F}_{rr}$, specifically $\|x_i(t) - x_j(t)\| \leq 2\delta_r$, where δ_r is the maximum physical extent of the robot from its reference frame. Define then $\mathcal{F}$ as the set of formations in which there are no collisions with obstacles in the environment, namely $\mathbf{x}(t) \notin \mathcal{F}_{r0}$, and there are no inter-robot collisions, namely $\mathbf{x}(t) \notin \mathcal{F}_{rr}$,

$$\mathcal{F} = \mathbb{R}^{2N} \setminus \mathcal{F}_{ro} \setminus \mathcal{F}_{rr}. \tag{2.1.5}$$

The definition of feasible formations in (2.1.5), called the configuration space, combined with the integral model of robots motion in (2.1.4) allows us to construct the following optimization

problem to find control inputs $\dot{\mathbf{x}}(t)$, that result in task completion,

$$\begin{aligned} \min_{\dot{\mathbf{x}}(t)} \quad & \Gamma\left(\mathbf{x}\left(t_f\right)\right) \\ \text{s.t.} \quad & \mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} \dot{\mathbf{x}}(s)ds, \quad \mathbf{x}(t) \in \mathcal{F},\ t \in [t_0, t_f]. \end{aligned} \tag{2.1.6}$$

The solution to (2.1.6), is a series of control inputs that result in a final formation $\mathbf{x}(t_f)$ that minimizes the task function $\Gamma(\mathbf{x}(t))$, while adhering to the kinematics of the robots and avoiding collision between robots and obstacles in the environment.

2.1.1 Communication Network

In order to provide situational awareness, the robots must not only maneuver safely through the environment while completing the task, but must also maintain a minimum Quality of Service (QoS) level. The QoS we consider is end-to-end data rate, namely what is the rate at which a robot can transmit its sensor measurements to any other member of the team via the ad-hoc network. For this we define K QoS requirements, one for each flow of sensor data. Note that we define the QoS as a flow of sensor data, this is done to allow for a single set of sensor measurements to be transmitted to multiple destinations. For the k^{th} QoS requirement we define the minimum end-to-end data rate that robot i must maintain for sensor data flow as $a^k_{i,m}$. Since each sensor data flow has an origin and a destination, we define $\mathcal{S}_k$ as the source and $\mathcal{D}_k$ as the destination of flow k. Since only the source of the data flow requires bandwidth, $a^k_{i,m}$ is non-zero for $i \in \mathcal{S}_k$, and $a^k_{i,m} = 0$ for $i \notin \mathcal{S}_k$. As an example, assume there is only one sensor data flow of interest, $K = 1$, originating from robot $i = s$ destined for the access point, $i = N$. With only one flow, $k = 1$ we would set $a^1_{s,m}$ to the data rate necessary to support the sensor and all other $a^1_{i,m} = 0$. We would also set $\mathcal{S}_1 = \{s\}$ to indicate robot s being the source, and $\mathcal{D}_1 = \{N\}$ to indicate that the access point $i = N$ is the destination.

To model how the location of the robots affects the underlying communication network we

begin with a normalized point-to-point rate function $R(x_i(t), x_j(t)) : \mathbb{R}^4 \to [0, 1]$. This measures the rate at which a robot i located at $x_i(t)$ can transmit data to robot j located at $x_j(t)$. By computing the rate between every pair of locations in the formation, $\mathbf{x}(t)$, we can compute a rate matrix $\mathbf{R}(\mathbf{x}(t)) \in \mathbb{R}^{N\times N}$, where element $R_{ij}(\mathbf{x}(t)) = R(x_i(t), x_j(t))$. While it is common to assume that $\mathbf{R}(\mathbf{x}(t))$ is symmetric, i.e. $R_{ij}(\mathbf{x}(t)) = R_{ji}(\mathbf{x}(t))$, due to channel reciprocity, we do not require this assumption to allow for extension to heterogeneous transmitters.

The data flow over the network is specified by a set of routing variables $\alpha_{ij}^k(t) \in [0, 1]$ which indicate the fraction of time that robot i is sending data to robot j for sensor data flow k. Similar to the rate matrix we collect the routing variables into a routing matrix $\boldsymbol{\alpha}(t) \in \mathbb{R}^{N\times N\times K}$ with entries $\alpha_{ij}^k(t)$. Since $\alpha_{ij}^k(t)$ is a fraction of time robot i is transmitting to robot j for sensor data flow k, the sum over all j and k must not exceed 1, $\sum_{j=1}^{N}\sum_{k=1}^{K}\alpha_{ij}^k(t) = \sum_{j,k}\alpha_{ij}^k(t) \leq 1$, for all i. Using this definition we can now compute the rate of data flow over the communication link from robot i to robot j for sensor data flow k as $\alpha_{ij}^k R(x_i(t), x_j(t))$. The amount of data flowing out of robot i for sensor data flow k can then be computed as $\sum_{j=1}^{N}\alpha_{ij}^k(t)R(x_i(t), x_j(t))$ and the amount of data flowing into robot i for sensor data flow k can be computed as $\sum_{j=1}^{N}\alpha_{ji}^k(t)R(x_j(t), x_j(t))$. If we consider the difference between the outgoing and incoming sensor data flows, minus the data destined for robot i, we can compute a communication rate margin,

$$a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) = \sum_{j=1}^{N}\alpha_{ij}^k(t)R(x_i(t), x_j(t)) - \sum_{j=1,\, i\notin\mathcal{D}_k}^{N}\alpha_{ji}^k(t)R(x_j(t), x_i(t)). \tag{2.1.7}$$

The exclusion of $i \in \mathcal{D}_k$ in the second summand of (2.1.7) captures the understanding that data destined for robot i does not impact the communication rate margin. This is because upon reaching i, the destination, there is no more need to relay the data and it can be removed from consideration. To provide network stability and prevent unbounded accumulation of data at a single robot, we require that $a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq 0$ for all i and k. This constraint allows us to reinterpret $a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$ as the amount of data that robot i can add to the network for sensor

data flow k without compromising network stability. Combining (2.1.7) with the QoS requirements for a data flow, we define the concept of network integrity, which is achieved when,

$$a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k, \qquad \sum_{j,k} \alpha_{ij}^k(t) \leq 1, \qquad \text{for all } i,\ k. \tag{2.1.8}$$

We see that when (2.1.8) is satisfied for $t \in [t_0, t_f]$, real-time situational awareness is achieved for the duration of the deployment.

Incorporating the network integrity constraints with the motion planning problem in (2.1.6) creates the concurrent mobility and communication problem,

$$\begin{aligned} \min_{\dot{\mathbf{x}}(t), \boldsymbol{\alpha}(t)} \quad & \Gamma\left(\mathbf{x}\left(t_f\right)\right) \\ \text{s. t.} \quad & \mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} \dot{\mathbf{x}}(s) ds, \quad \mathbf{x}(t) \in \mathcal{F}, \qquad \text{for all } t \in [t_0, t_f], \\ & a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k, \quad \sum_{j,k} \alpha_{ij}^k(t) \leq 1, \qquad \text{for all } i,\ k. \end{aligned} \tag{2.1.9}$$

The problem in (2.1.9) seeks to find optimal motion control inputs, $\dot{\mathbf{x}}^*(t)$, and routing variables, $\boldsymbol{\alpha}^*(t)$, that not only allow the team to reach $\mathbf{x}(t_f)$ that completes the task, $\Gamma(\mathbf{x}(t_f)) = \Gamma_{min}$, but also maintain network integrity for the duration of the deployment.

2.1.2 Channel Estimation and Network Routing

Considering only the network routing portion of (2.1.9) we notice the formulation interprets the link rates as known deterministic quantities, which is not true for wireless links. As shown in [52] wireless links are best modeled as independent random variables, where the value $R(x_i(t), x_j(t))$ has a mean $\bar{R}(x_i(t), x_j(t)) := \mathbb{E}\left[R(x_i(t), x_j(t))\right]$ and variance $\tilde{R}(x_i(t), x_j(t)) := \text{var}\left[R(x_i(t), x_j(t))\right]$. Due to the interpretation of the rates as random variables, the end-to-end data rates, $a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$, also become random variables, with mean $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) := \mathbb{E}\left[a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))\right]$ and variance $\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) := \text{var}\left[a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))\right]$. The relative ease, or complexity, of determining an expression for $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$

and $\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$ rely on what model is used for $R(x_i(t), x_j(t))$.

This reinterpretation of the end-to-end data rates as random variables introduces a delineation in the formulation of the routing problem. On one side of the line there is a set of formulations that only incorporates the expected value of the end-to-end data rate into the QoS constraints, while on the other side, the expected value and the variance, or a proxy for the variance, of the end-to-end data rate is incorporated into the QoS constraints.

When only considering the expected value of the end-to-end data rates the following optimization problem can be used,

$$\begin{aligned} \boldsymbol{\alpha}(t) = \underset{\boldsymbol{\alpha}(t), a_\Delta}{\operatorname{argmax}} \quad & a_\Delta \\ \text{s. t.} \quad & \bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k + a_\Delta \\ & \sum_{j,k} \alpha_{ij}^k(t) \leq 1, \end{aligned} \tag{2.1.10}$$

where the value a_Δ is a newly introduced slack variable. The maximization of a_Δ results in a routing solution in which the amount that each end-to-end data rate exceeds its minimum requirement is maximized. This formulation, and those similar to it, focus on maximizing throughput, which can be thought of as maximizing the upper bound of the end-to-end data rate.

The other set of formulations attempt to mitigate the variation in the end-to-end data rate by understanding that the difference between the realized channel and the expected channel may be large. Since knowledge of $\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$ can be difficult to obtain, due to insufficient information or the characteristics of $R(x_i(t), x_j(t))$, these formulations are further bifurcated into formulations that use the variance directly and those that employ a proxy for the variance. One such formulation

that utilizes a proxy for the variance is,

$$\begin{aligned} \boldsymbol{\alpha}(t) = \underset{\boldsymbol{\alpha}(t)}{\operatorname{argmax}} \; & -\sum_{k=i}^{K}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\alpha_{ij}^{k}\right)^{2} \\ \text{s.t.} \quad & \bar{a}_{i}^{k}(\boldsymbol{\alpha}(t),\mathbf{x}(t)) \geq a_{i,m}^{k}, \\ & \sum_{j,k}\alpha_{ij}^{k}(t) \leq 1. \end{aligned} \tag{2.1.11}$$

In (2.1.11) by maximizing the negative of sum of the square of the routing variables, the optimal routing solution will use as many communication links as possible while maintaining the required rate, which has been shown to minimize end-to-end variance. When the variance is available the formulations can be more strict in the requirements such as,

$$\begin{aligned} \boldsymbol{\alpha}(t) = \underset{\boldsymbol{\alpha}(t),a_{\Delta}}{\operatorname{argmax}} \; & a_{\Delta} \\ \text{s.t.} \quad & \mathrm{P}\left[a_{i}^{k}(\boldsymbol{\alpha}(t),\mathbf{x}(t)) \geq a_{i,m}^{k} + a_{\Delta}\right] > 1-\epsilon \\ & \sum_{j,k}\alpha_{ij}^{k}(t) \leq 1. \end{aligned} \tag{2.1.12}$$

In (2.1.12) the slack variable a_{Δ} again is maximized as in (2.1.10), but in this formulation the minimum end-to-end rate constraint is stated in a probabilistic sense. Instead of just maximizing the expected end-to-end data rates, now the constraints require that the probability of achieving that end-to-end data rates must meet a threshold. The threshold is controlled by the parameter ϵ which can be thought of as the user's acceptance of risk in regards to maintaining network integrity. If network integrity is critical and any loss would be catastrophic then ϵ is chosen to be very small. Conversely, if periods of lost network integrity are acceptable then ϵ can be larger. The result of (2.1.12) can be interpreted as maximizing the lower bound of the end-to-end data rate, which is preferable to maximizing the upper bound as done by (2.1.10). This is because the formulation in (2.1.12) provides a probabilistic guarantee on the minimum as opposed to an

optimistic estimate of the maximum.

While not as obvious as (2.1.11) the routing solutions from (2.1.12) also utilize a more diverse set of links than (2.1.10). Due to this property, the solutions that result from formulations such as (2.1.11) and (2.1.12) are referred to as robust routing solutions because, as the name suggests, they are robust to the failure of a single link. In contrast, the routing solutions that result from (2.1.10) are referred to as non-robust routing solution since a single link failure could cause the entire data flow to be interrupted. Through out this thesis we will make reference to and use modified versions of (2.1.10), (2.1.11), and (2.1.12).

2.2 System Architectures

In Section 2.1.1 we introduced a formulation of the concurrent mobility and communication problem that is central to the this thesis. This section provide a taxonomy of the previously proposed systems. By doing this, we are able to identify the benefits and limitations of each system, in order to design a system that can leverage the advantages of previous work, while avoiding the pitfalls.

2.2.1 Control Law

When designing a system architecture for mobile robot teams, the first decision to be made is whether to implement a local or global control law. A local control law, sometimes referred to as distributed, is a system design in which each robot on the team takes the optimal action given locally available information. Conversely, a global control law is a system design in which the actions for each robot are determined by a centralized compute node. This requires an aggregation of all the available information to determine the actions that will lead to an optimal outcome for the team as a whole.

Focusing on the systems that implement a local control law [18, 34, 70, 74, 96–98], we notice a

few commonalities. By only requiring local information to determine their actions, the amount of coordination overhead is minimal. This is due to the robots only sharing information with immediate neighbors, and not with those beyond. This attribute allows the team size to grow with minimal impact on required coordination because from the perspective of a single robot, the addition of other robots will only increase its coordination level if that robot is placed inside its communication range. Unfortunately, due to the robots acting with only local information, they are unable to avoid local minima, which are caused by the interaction of the complex environment and the communication constraints. While these systems have demonstrated the ability to successfully maintain specific communication metrics, the inability to avoid local minima limits the environments in which they are effective.

Next, we consider the systems that implement a global control law [21,23,24,39,81,83,95]. As with the local control law systems, some trends begin to emerge. Initially, we notice that these systems are able to operate in more complex environments than the local control systems. This is due to the centralized compute node determining the optimal action for the team as a whole instead of each individual robot acting independently. An example of this is highlighted in [24] where one of the robots must move in a direction counter to its locally optimal direction in order to provide support to a robot as it rounds a corner. This ability to determine globally optimal solutions comes at the cost of aggregating all of the information at a central location. While this system architecture performs well when the team sizes are small, as the number of robots on the team increases, so does the coordination overhead. Eventually, this overhead will be too burdensome and the system will be unable to operate effectively.

While the proposed systems can be used to satisfy requirements on connectivity, only the systems in [24, 98] specifically incorporate network routing into the problem formulation. These two systems approach the process of determining optimal network routes in the same manner that the motion control law is approached; [98] considers the routing problem in a distributed manner while [24] approaches the problem in a global manner. It is reasonable that the network routing

		Solution Optimality	
		Local	*Global*
Coordination Approach	*Distributed*	[18, 34, 70, 74, 96–98]	*
	Centralized	-	[21, 23, 24, 39, 81, 83, 95]

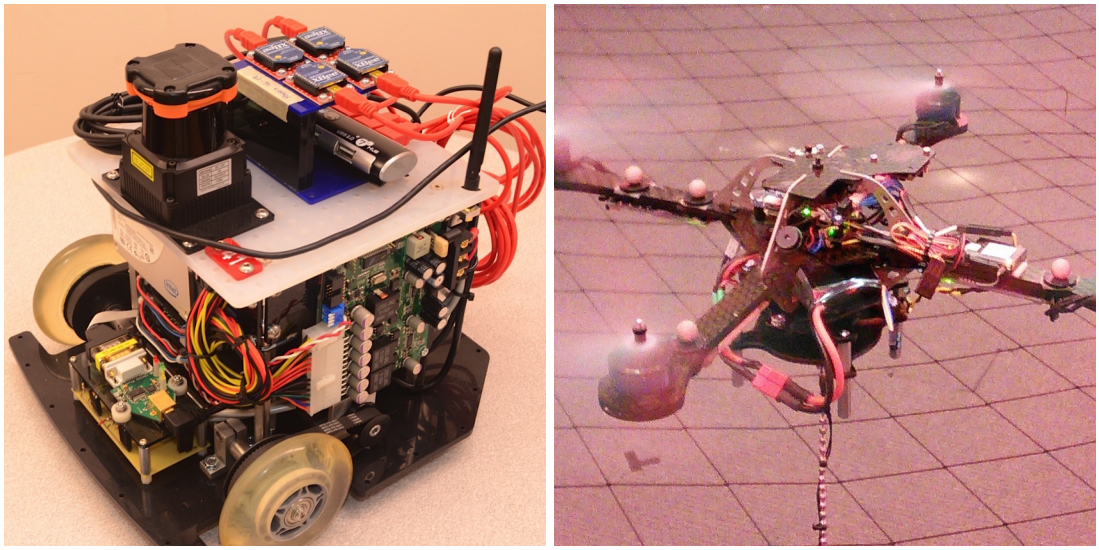
Table 2.1: System Design Matrix with current systems placed according to where they reside in the 2-D system design space.

approach would mimic the motion control law, since if global information is available for motion planning it should also be used for network routing. Therefore, we can classify the prior systems into two main classes, centralized and distributed.

2.2.2 System Design

With an understanding of previous systems, we can begin to draw conclusions as to how best architect a system that can meet all the requirements of real-time situational awareness. As highlighted in Section 2.2.1, the two classes of systems proposed, centralized and distributed, each suffer from inherent drawbacks. This leads us to reinterpret system design space as a two-dimensional space, as opposed to the one-dimensional space typically used. A representation of the design space is shown in Table 2.1, where the two axis represent coordination approach, distributed or centralized, and optimality of solution, local or global. By viewing the design space in this manner we see that all of the proposed systems fall in two of the four segments. The distributed systems all employ distributed coordination, but are limited to local optimality, while the centralized systems are able to achieve global optimality at the cost of centralized coordination.

The two remaining segments, are empty for different reasons. First, the lower left corner is empty because it is a sub-optimal segment of the space. The reason for this is if a system resides in this segment it is requiring centralized coordination to provide a locally optimal solution. It would be more efficient to operate in a distributed manner and remove the coordination overhead, or to



(a) The newest generation of the *Scarab*s. The XBees are mount on top of the platform behind the Hokuyo.

(b) The quad-rotor platform that is used in the hallway monitoring demonstration. The XBee is mounted on the underside.

Figure 2.1: Robotic platforms.

use the centralized coordination to determine a globally optimal solution. Therefore, any system that resides in this segment will shift either up or to the right, joining the current distributed or centralized systems.

The upper right corner is still empty because a system that can provide globally optimal solutions to the concurrent mobility and communication problem in arbitrarily complex environments with distributed coordination has yet to be proposed. This is the optimal segment of the space, as indicated by the *, due to the fact that a system in this segment is able to satisfy the requirements for real-time situational awareness, as well as many other open problems in multi-robot systems. Specifically, It would be able to operate in increasingly complex environments without concern of local minima, while scaling efficiently as the team grows larger.

2.3 System Implementation

2.3.1 Hardware

For this thesis, we primarily use a team of *Scarab*s [48], a custom built robot designed at the University of Pennsylvania, as our robotic platform. The 4^{th} generation of the *Scarab* includes onboard computing, a Hokuyo UTM-30LX scanning laser range finder with a 30 meter range, and two Robo Claw 5 amp Motor Controllers. The motors are used to drive two of the three wheels, while the third is a passive omni-directional wheel. Since the *Scarab* is a small differential-drive platform, it is straight-forward to apply feedback linearization in order to obtain appropriate control inputs given the kinematic control laws presented in this thesis. The on-board computer contains an Intel i7 3.4 GHz processor, 8 GB of RAM, and a 250 GB SSD hard drive with a full installation of Ubuntu 14.04 LTS. An image of a standard *Scarab* can be seen in Fig. 2.1a.

For wireless communication between *Scarab*s we use the Digi International XBee transceivers [90]. These modules allow the user to control frequency and power. The XBee radios are capable of transmission on 16 evenly spaced channels in the 2.4 GHz spectrum. The XBee radio also allows for 5 discrete power levels, ranging from -10 dBm to 0 dBm. The XBee transmits data via a fixed packet size of 100 bytes, with a preamble the result is an effective payload size of 90 bytes for each transmission.

As shown in Fig. 2.1a, each *Scarab* in these experiments contains 4 XBees. Each XBee is configured to transmit at the minimum power of -10 dBm to force reliance on the other robots on the team while keeping the size of experiments manageable. Additionally, each XBee is responsible for communication on a different frequency. The frequencies chosen are evenly spaced to allow for maximum signal isolation between radios. This allows for the communication between one pair of *Scarab*s to not interfere with communication between another pair of *Scarab*s, which is important as our formulation does not consider interference. As demonstrated in [22] there exist signal models that can effectively predict the expected signal prorogation for the low power XBees

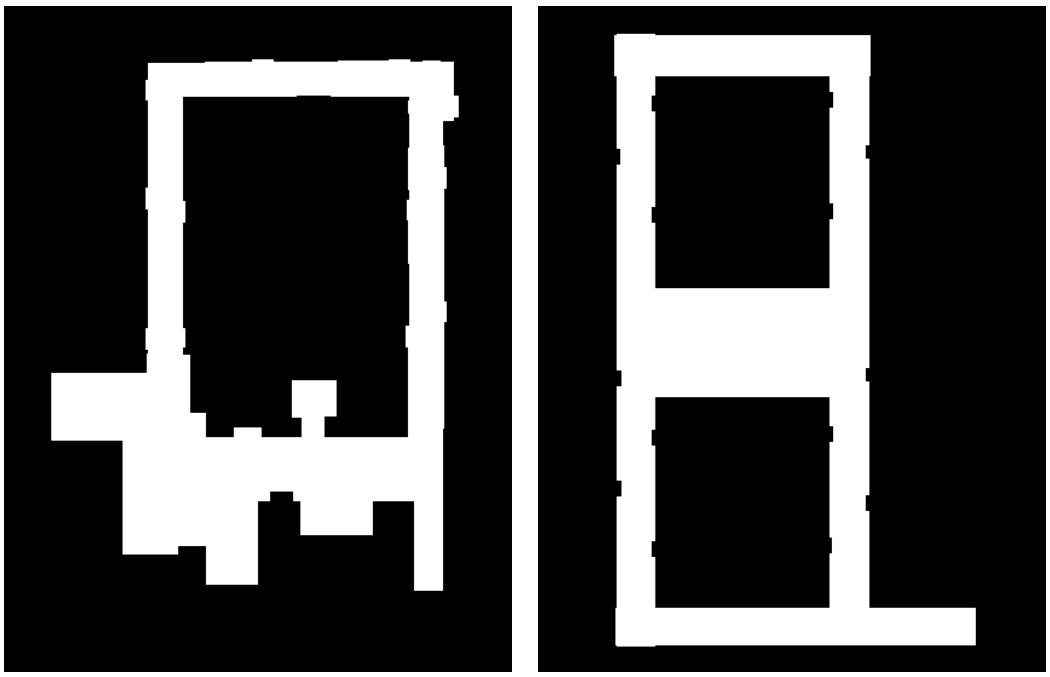
used in this thesis.

In addition to the *Scarab* platform we utilize a quad-rotor, show in Fig. 2.1b. The specific quad-rotor model used is the Hummingbird from Ascending Technologies [5] equipped with an Overo Gumstix for on-board control and a single XBee transmitter. This robot is used in experiments where a highly-mobile platform is needed to move through the environment.

2.3.2 Software

For this thesis we rely on the Robotic Operation Systems (ROS) framework [63], to facilitate software development, simulation, and experimentation. The ROS framework consists of a middleware that abstracts the hardware layer and standardizes the interface to processing algorithms. In ROS all components of the system are referred to as nodes, whether they are responsible for interfacing with sensors or executing high level processing algorithms, with a central coordination node called the master. This configuration allows for extensive modularity and portability, meaning a processing algorithm developed in the ROS framework for one sensor payload can be easily modified and used on a different sensor payload. This modularity has allowed for the creation of a large repository of algorithms common to robotics, maintained mostly by Willow Garage, [88]. Additionally, since ROS is a middleware, it simplifies the process of software development from initial simulations to final experimentation. This is enabled by the ability to begin in a full simulation environment, then slowly replace simulation nodes with hardware, until all of the processing and sensing is operating on hardware. Finally, since ROS is designed to operate as a collection of discrete nodes, it is easily extensible to provide coordination over teams of robots. This also facilitated the software debugging process because even though processing and control is occurring on the robot, the master node is always able to interrogate the state of the robot.

The primary nodes used in the system other than those developed by the Multi-Robot Systems Laboratory (MRSL) at the University of Pennsylvania [53], are *AMCL* an Adaptive Monte Carlo Localization library for estimating the location of the robots in known environments, and *gmapping*



(a) Levine 5^{th} floor. (b) Levine-GRW 5^{th} floor.

Figure 2.2: Maps of the location where most of the experiments take place.

a Rao-Blackwellized particle filter for Simultaneous Localization and Mapping (SLAM) for use in unknown environments.

2.3.3 Environments

Through out this thesis a a variety of experiments are performed in a series of environments. All of the experiments are performed within two of the engineering buildings at the University of Pennsylvania. The two primary locations that are used throughout this thesis are shown in Fig. 2.2. The area on the left is the 5^{th} floor of the Levine building and the area on the right is the 5^{th} floor of the Graduate Research Wing referred to as Levine-GRW. The other environments used in this thesis are detailed in the chapter in which they appear.

2.4 Summary

In this chapter we begin by posing the real-time situational awareness task as a concurrent mobility and communication problem. We then discuss the various formulations of network integrity used in the mobility and communication problem, specifically a non-robust and two robust formulations. Next, we survey the design space in which solutions to the concurrent mobility and communication problem reside. We interpret the problem of optimal system design as 2-dimensional, as opposed to the typical 1-dimensional, problem. The two dimensions of this space are the level of coordination required, and the optimality of the resulting solution. With this interpretation we determine the desirable properties for a system in regards to the situational awareness task, namely globally optimal solutions achieved though distributed coordination. We conclude the chapter with an overview of the robotic platforms, software framework and primary environments used throughout this thesis.

Chapter 3

Multi-Confirmation Transmission Protocol

The formulation of the concurrent mobility and communication problem in (2.1.9) requires the computation of routing variables $\boldsymbol{\alpha}(t)$ by which data is moved through the ad-hoc network. The determination of these $\boldsymbol{\alpha}(t)$ can vary in complexity from selecting the sequence of links that has the highest minimum rate and sending all data along that path, to spreading the data over as many different paths as possible. While the former approach is commonly taken in wired networks, due to the rate of a point-to-point link being easy to predict, the same approach in a wireless network results in large variations in the experienced end-to-end rate. This is due to the random fluctuations experienced by the wireless link; as such there has been research into optimal network routing over wireless networks, [66, 67, 89]. As highlighted in Section 2.1.2, routing solutions that consider the variations in the end-to-end data rate as well as the expected value are referred to as robust routing solutions, while those that only consider the expected value are called non-robust.

In this chapter we demonstrate the performance advantages of using a robust routing solution over a non-robust one in the context of real-time situational awareness. Then, we will examine

the feasibility of using existing communication protocols over an ad-hoc wireless network. We will subsequently construct a communication protocol that is designed for the robust routing solutions and low-power transceivers used in mobile robot communication networks. Finally, we demonstrate the benefits of our protocol to provide real-time situational awareness. Content from this chapter originally appeared in [77].

3.1 Non-Robust vs Robust Routing

To examine the effect that the choice of routing solution has on the experienced end-to-end data rate we begin with a series of experiments. The objective of these experiments is to validate the assumption that a robust routing solution is preferable to a non-robust routing solution for our application, as well as provide a baseline for system performance. In these experiments the task is to drive the robots through the environment so that a designated robot is able to reach a specific location. This can be thought of as a simple situational awareness task, given a team of robots how can they self configure so that a single sensing robot is able to obtain measurements at a specific location.

To determine the trajectory and control the motion of the robots we utilize the centralized system from [24]. In this system the trajectories are generated a priori to guarantee network integrity for the duration of the deployment and the motion of the robots is strictly controlled by the centralized node. This provides a high level of control over the evolution of the formation, and thus repeatable trajectories for experimental comparisons.

The first class of routing solutions that we consider are non-robust solutions. We use the

formulation in (2.1.10),

$$\begin{aligned}\boldsymbol{\alpha}(t) = \underset{\boldsymbol{\alpha}(t), a_\Delta}{\text{argmax}} \quad & a_\Delta \\ \text{s.t.} \quad & \bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k + a_\Delta \\ & \sum_{j,k} \alpha_{ij}^k(t) \leq 1,\end{aligned}$$

for our example routing optimization problem. When solving this problem the system will attempt to maximize the amount of margin each node can obtain over their minimum requirement. This approach is used instead of a greedy routing solution that maximizes a single node's margin, because the excess communication margin is dispersed evenly over the team instead of concentrated at one location; thus maximum fairness is achieved.

For the robust routing formulation we use (2.1.12),

$$\begin{aligned}\boldsymbol{\alpha}(t) = \underset{\boldsymbol{\alpha}(t), a_\Delta}{\text{argmax}} \quad & a_\Delta \\ \text{s.t.} \quad & \mathrm{P}\left[a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k + a_\Delta\right] > 1 - \epsilon \\ & \sum_{j,k} \alpha_{ij}^k(t) \leq 1.\end{aligned}$$

This formulation will result in a solution that achieves the maximum margin, as in the non-robust solution but with a higher confidence. This is precisely the logic behind robust routing, a small decrease in the maximum achievable rate is allowed to provide more confidence of that the minimum will be maintained.

3.1.1 Experimentation

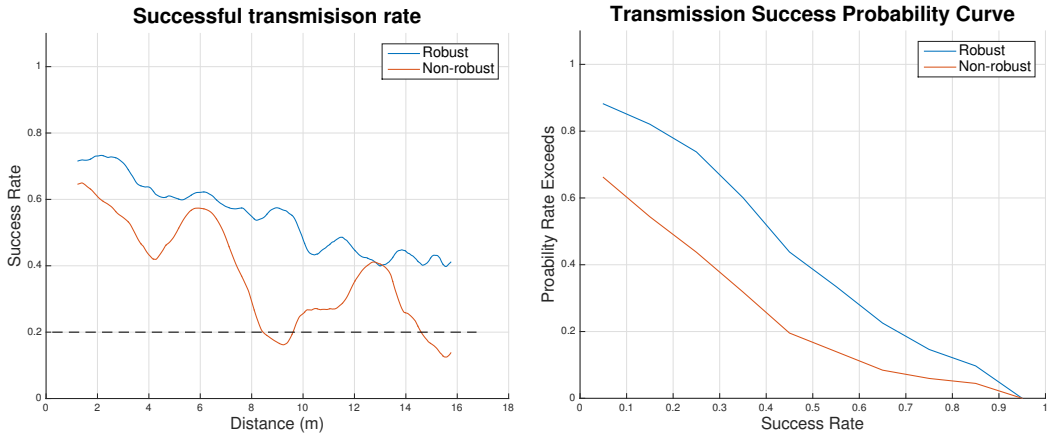
To compare non-robust and robust routing we ran a series of experimental trials, in which the team of robots is tasked with supporting a single sensing robot as it moves to a desired location.

As noted in the previous section, we use a centralized system to closely control the motion and routing, so that a comparison of the routing solutions can be performed. In these trials the robots follow the same prescribed path and use the same channel estimation parameters regardless of which routing solution is used. Additionally, the routing solutions are computed and provided to the team at the same rate. This will allow for an unbiased comparison of the performance of both solutions.

Levine Environment

In the first series of experiments the 5^{th} floor of the Levine building at the University of Pennsylvania was used (Fig. 2.2a). In this experiment, the team consisted of 3 mobile robots and a fixed access point, which begin in the open area in the lower left hand corner of the environment. This series of experiments consisted of three trials for each routing formulation. The trajectory followed is validated against the robust and non-robust solution to confirm feasibility. Since there is only one robot requiring communication back to the access point $a^1_{i,m} = 0$ for all i except for robot $i = 3$, for which we set $a^1_{3,m} = 0.2$. For the robust formulation we model the variance of the channel as a Gaussian random variable with zero mean and variance $\sigma^2 = 32\text{dB}^2$, and require an 80% probability of satisfying the constraints, or $\epsilon = 0.2$.

For these experiments, we use the rate of successful packet reception at the access point as a proxy for the end-to-end data rate. Also, in these experiments, UDP is used so that the performance of the routing solution is not obfuscated by any retransmissions provided by the transport layer. This information well help inform the construction of the communication protocol in the later section. As the robots move through the environment, the optimal routing solution is computed based on the estimated robot locations and transmitted over the ROS coordination network. The sensing robot, $i = 3$, generates data at a rate of 1 kilobyte per second, or 10 packets per second. The speed of the robots is restricted to 10 cm/sec so that an accurate measure of the end-to-end rates can be determined.



(a) Percentage of successfully received packets using UDP. The robust solution provides more reliable performance while the non-robust solution varies greatly.

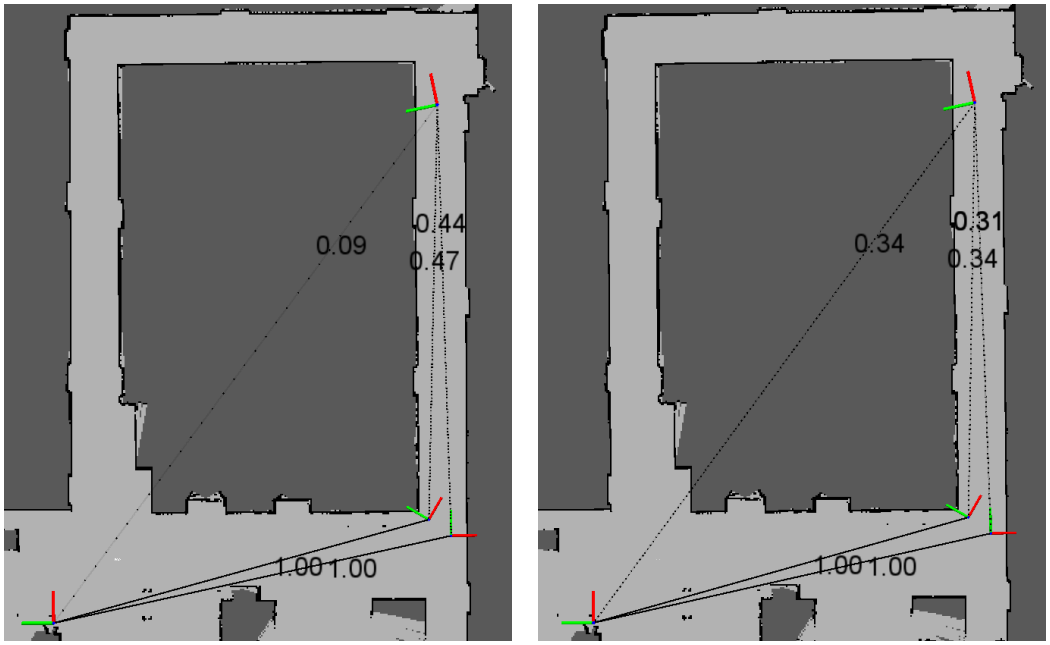
(b) The probability that the routing solution will provide a data rate greater than or equal to the requested value.

Figure 3.1: Levine 5^{th} floor experiments, $N = 4$.

The trajectory that is executed drives the lead robot from the lower left hand corner to the right until it reaches the right side hallway, at which time it turns and proceeds up the hallway to the intersection in the top right corner. This trajectory covers a total of 23 meters and results in a final separating distance from the access point of 16 meters. In unison, the other two robots move from the lower left hand corner to the lower right hand corner and assume positions that provide line of sight to both the sensing robot and the access point.

The average end-to-end data rate experienced by the sensing robot back to the access point for both the non-robust and robust solutions are shown in Fig. 3.1a. In this figure the average end-to-end data rate is plotted as a function of distance between the sensing robot and the access point. There are two items of note, the first of which is that the robust solution outperforms the non-robust solution for the entirety of the trial. Another item to note is the relative stability of the robust solution; as expected the robust solution hedges against variance, therefore resulting in a smoother degradation of the data rate. Specifically, the standard deviation over the path for the robust solution is 0.1081 in comparison to the non-robust solution which had a value of 0.1536.

In Fig. 3.1b we plot the probability of exceeding the given rate based on the empirically



(a) Final formation with routing solution for non-robust system in Levine.

(b) Final formation with routing solution for robust system in Levine.

Figure 3.2: Non-robust (left) and robust (right) routing solution for the 5^{th} floor Levine. The non-robust solution relies heavily on the line-of-sight paths, this allows for maximum data rates but does not mitigate fading.

collected data. To interpret this figure, the x-axis is a specified success rate and the y-axis is the probability of exceeding that success rate. By plotting the empirical data we can see the probability of exceeding a given success rate. Therefore, the closer the line is to the upper right corner the better the systems are performing. The first important item from this plot is that the robust routing solution has a 0.78 probability of exceeding a data rate of 0.2. This closely matches the input parameters of $a^1_{3,m} = 0.2$ and $1 - \epsilon = 0.8$. Also note that, even though the formulations were done with $a^1_{3,m} = 0.2$, the resulting robust and non-robust routing solutions provide data rates that exceed 0.4 with probability 0.52 and 0.25, respectively. In this scenario it is obvious that the robust routing formulation greatly outperforms the non-robust routing.

To highlight the differences in the routing solutions we show in Fig. 3.2 the environment as well as a final formation for one of the experimental trials. In the figure we see that for the same formation the non-robust (Fig. 3.2a) and the robust routing (Fig. 3.2b) solutions differ.

Specifically, the ratio of the data being sent back directly to the access point is 0.09 for the non-robust solution and 0.34 for the robust solution. This vast difference is due to the fact that the robust solution takes into account that there is an equal probability of a successful transmission when using the link that connects directly back to the access point, as there is when using the two-hop link that connects through either support robot.

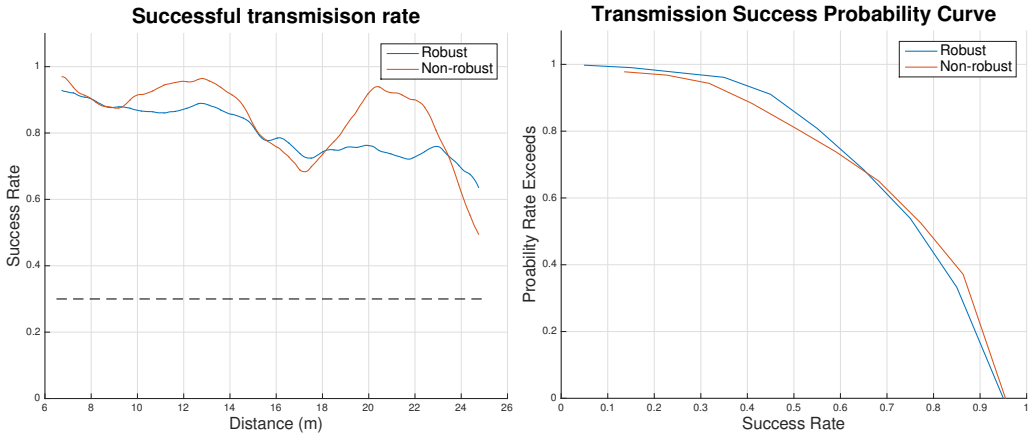
Levine-GRW Environment

In this section, we detail another experiment that compares robust routing to non-robust routing. The purpose of this experiment is to confirm the benefits of robust routing and show that they are not limited to one environment.

This set of experiments takes place on the 5^{th} floor of Levine-GRW (Fig. 2.2b) and consists of two runs for each routing solution. The path planning portion of this experiment does not use the centralized method, but instead is performed manually in order to optimize the number of line-of-sight links. This experiment uses the same configuration as the previous set expect for the following modifications. The team now consists of 4 mobile robots and a fixed access point, with robot $i = 4$ now performing the sensing task with a minimum rate of $a^1_{4,m} = 0.3$. With the increase in the minimum rate the probability threshold is relaxed to $\epsilon = 0.25$ resulting in a 75% confidence of achieving the rate. The routing solutions are again computed in real-time and transmitted to the robots with UDP used as the communication protocol.

In this experiment set, the 4 robots and the access point begin in the lower left corner of the environment. In this experiment, the sensing robot moves to the right along an 11 meter hallway until reaching the intersection. It then turns left and travels up the 25 meter hallway, eventually reaching the top right corner of the environment. Meanwhile, 3 support robots move in order to support the minimum data rate requirement.

The average end-to-end success rate is plotted as a function of distance in Fig. 3.3a. In the figure we can see that the non-robust routing solution consistently outperforms the robust solution.



(a) Average ratio of successfully received packets using UDP. The robust solution provides more reliable performance while the non-robust solution varies greatly.

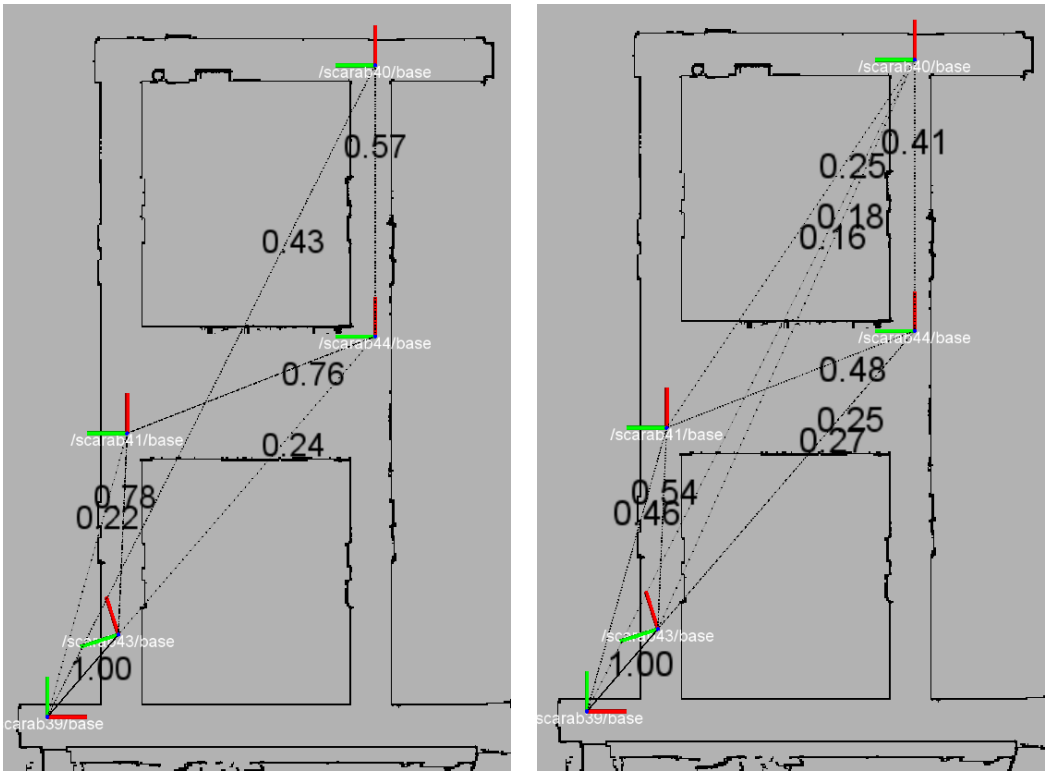
(b) The probability that the routing solution will provide a data rate greater than or equal to the requested value.

Figure 3.3: Levine-GRW 5^{th} floor experiments, $N = 5$.

This is in contrast to the previous results. Also, the rapid fluctuations are again present in the non-robust routing solution, but not in the robust routing solution. Additionally, the robust routing solution has a lower standard deviation in comparison to the non-robust solution, which are 0.0837 and 0.120, respectively.

The over performance of the non-robust solution in this experiment set, when compared with the results in Fig. 3.1a, is due to the routing solutions optimizing different quantities. As stated in Section 2.1.2 the non-robust solution is optimizing for maximum throughput while the robust solution is attempting to mitigate the variations in the end-to-end rate. Taking a closer look at Fig. 3.3a we see in fact that both systems performed as expected. The robust solution minimized end-to-end variations and the non-robust maximized throughput.

In Fig. 3.3b we again plot the probability of exceeding a given data rate based on the empirical data collected. The first important result from this plot is that the robust routing solution has a 0.96 probability of exceeding a data rate of 0.3. This far exceeds the input parameters of $a^1_{4,m} = 0.3$ and $1 - \epsilon = 0.75$. Also note that the robust routing solution provides a higher probability for a given data rate, up until 0.65, at which point the large fluctuations assist the non-robust solutions



(a) Final formation with routing solution for non-robust system in Levine-GRW.

(b) Final formation with routing solution for robust system in Levine-GRW.

Figure 3.4: Non-robust (left) and robust (right) routing solution for the 5^{th} floor GRW. The non-robust solution relies heavily on the line-of-sight paths, this allows for maximum data rates but does not mitigate fading.

data rates.

In Fig. 3.4 we plot the final formation of the team along with the routing solution for the non-robust and robust solutions. In the figure we see that for the same formation the non-robust (Fig. 3.4a) and the robust routing (Fig. 3.4b) solutions again differ. The differences are similar to those in Fig. 3.2, most notably the increased number of links out of the sensing robot in the robust solution compared to the non-robust solution.

3.2 Protocol Design

As the results from the experiments demonstrate, the benefit of utilizing a robust routing solution outweighs the minor benefit of higher throughputs. Therefore, we focus on how the current standard communication protocols behave over ad-hoc wireless links that utilize robust routing solutions.

3.2.1 Current Protocols

The two ubiquitous protocols in use today are UDP and TCP. The inability of UDP to provide confirmation of packet reception is a strong inhibitor to using it as our protocol, but the high channel utilization that it demonstrates is desirable. The other option is TCP, whose sub-optimal performance over wireless networks is extensively researched, [3, 9, 58, 59]. Even with the extensive array of modifications to the protocol as documented in [3], these solutions are narrowly tailored to a specific scenario and network topology. Additionally, with the introduction of the robust routing, no longer is the data following along a stable deterministic path which is required for optimal TCP performance.

With the realization that neither of the two common communication protocols are sufficient for our system, we wish to design a lightweight communication protocol that is capable of achieving the high link efficiency seen in UDP, while providing the confirmation of reception as in TCP. This protocol must also be able to operate on the low-power low-cost transceivers and leverage the link diversity present in robust routing solutions.

3.2.2 MCTP

These requirements lead to the development of the Multi-Confirmation Transmission Protocol (MCTP). MCTP is a modification of Nagle's algorithm for small packets [54] and takes advantage of the spatial redundancy by allowing a packet that failed over one link to be retransmitted over

Algorithm 1 Algorithm for receive

Require: number of robots N, maximum number of packets in confirm message P, number of new message to respond to M, and maximum time between responses T

```
1: Initialize N queues {Q_i} and timers {T_i}
2: while System running do
3:    if Packet p successfully received from robot s then
4:       add p_id to Q_s
5:       if size of Q_s mod M equals 0 then
6:          Send the contents of Q_s to s, reset T_s
7:       end if
8:       if size of Q_s > P then
9:          Pop oldest element off of Q_s
10:      end if
11:   end if
12:   for i = 1 to N do
13:      if T_i > T then
14:         Send the contents of Q_i to i, reset T_i
15:         Pop off Q_i mod M oldest elements of Q_i
16:      end if
17:   end for
18: end while
```

a different link. This reduces the likelihood that a packet will be lost when one link is removed, as subsequent retransmissions of the packet will not use that particular link. This approach mitigates the random losses that occur over wireless channels due to link failure. Therefore, this communication protocol combines the benefits of both TCP and UDP protocols, to allow for efficient and reliable communication over a multi-hop wireless network.

The protocol operates in the following fashion. Initially, each robot is assigned a unique identifier, s, for this implementation we use, $s \in \{1 \dots N\}$. When there is data to transmit, we uniquely label and send the packet according to the routing solution C_i. The routing solution is expressed as a unique CDF for each robot, i, with the mass on j proportional to the percentage of data i should send to j. Therefore, to determine the destination, t, that robot i should use for each packet a random variable, x, is drawn from a uniform distribution from 0 to 1 and t is determined such that $x \in C_{i,t}$. We then insert the label into a data structure, $\mathbb{A}$, to record that the packet has been sent, but successful transmission has not been confirmed. We repeat this process for all outgoing packets.

Algorithm 2 Algorithm for transmit

Require: number of robots N, timeout value S, number of retransmission allowed R, and routing solution CDF, C_i

```
1: Initialize map T and A
2: while System running do
3:    if Packet p to send then
4:       Draw random value x from U[0, 1]
5:       Transmit p to robot t for which x ∈ C_{i,t}
6:       Add t_now to T and 1 to A for id of p
7:    end if
8:    if Confirmation packet received then
9:       for Each id in packet payload do
10:         Remove id from T and A
11:      end for
12:   end if
13:   for all (j, t) in T do
14:      if t < current time - S then
15:         if A[j] < R then
16:            Draw random value x from U[0, 1]
17:            Transmit p to robot t for which x ∈ C_{i,t}
18:            Update T[j] with t_now and increment A[j]
19:         else
20:            Remove mapping for j from T and A
21:         end if
22:      end if
23:   end for
24: end while
```

Upon successful reception of a packet, we add the label to a data structure, $\mathbb{B}$, and respond with the last P labels, but only if there have been M new packets since last response. Upon receiving a confirmation message, we compare the labels to those in $\mathbb{A}$. If a label in the confirmation response matches one in $\mathbb{A}$, the label is removed and successful transmission of the packet is confirmed. If, however, after a specified T seconds transmission is not confirmed, the packet is re-sent. This process can be seen in Algorithms 1 and 2. The values of T, P and M can be changed to provide the desired level of delivery guarantee depending on the scenario.

In order to allow for successful reassembly at the receiving side, each message, of arbitrary length, that is to be transferred is assigned a unique 16-bit identifier. Then each message is broken up in to 85-byte chunks, suitable for transmission, and assigned a number. The message identifier as well as the chunk number are added to the payload of the transmitted packet. These

packets are then provided as input to the MCTP algorithm, specifically line 3 for Algorithm 2. This information is then used to reassemble the message at the final destination as well as prevent re-transmission of duplicate packets inside the mulit-hop network. MCTP also leverages the redundancy of the routing layer by allowing a packet to travel multiple paths to the final destination with only minimal overhead. This is due to the random selection of the link to use during re-transmission. Therefore, as the amount of redundancy in the routing solution increases the more diverse set of paths a packet is able to traverse and the more robust the system becomes.

By allowing the sender to continue transmitting packets without waiting for a response, similar to TCP, the channel utilization is higher than if the sender waited for transmission confirmation for each packet. However, by using a fixed response window size of M and not a dynamic window size, such as in TCP, channel utilization again increases.

3.3 Protocol Validation

In order to validate that MCTP provides reliable communication over an ad-hoc wireless network we perform a series of experiments comparing the communication reliability of MCTP compared to UDP and Simple ACK. In these experiments the parameters used for MCTP are $M = 5$, $T = 0.5$ sec, $S = 1.0$ sec, and $R = 4$, with data again generated at 10 Hz. For a comparison we use Simple ACK, a lightweight protocol in which a response is given for every packet received. This mimics a configuration where over every link a TCP session is present. For Simple ACK the timeout between retransmissions and number of retries is the same as the values used in MCTP.

3.3.1 Protocol Experiment

The setup for this experiment involves a sensing robot moving away from the access point and traveling through the environment where direct line-of-sight is not always possible. Specifically the robot is operating in Levine-GRW with the same initial location as in the previous test. The

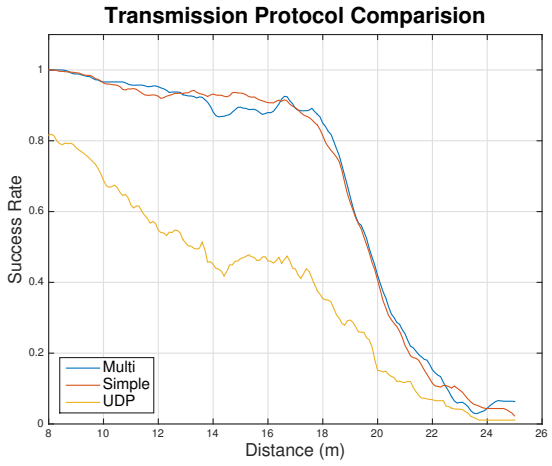


Figure 3.5: Plot for a single robot moving away from the access point and turning a corner at 11 meters using three transmission schemes. As expected the two schemes that utilize confirmations show much more reliable packet transmission. Note that MCTP achieves the same performance as Simple ACK with 5 times less confirmation packets transmitted.

robot starts by moving away from the access point and turns a corner, 11 meters away, before continuing to the end of the hallway. Without other robots to assist in the packet routing, this experiment tests the performance of point-to-point communication, $N = 2$ and $C_1 = 1.0$. Three experimental sets are run one for each protocol, UDP, Simple ACK and MCTP, with three trials per protocol.

It can be seen in Fig. 3.5 that for a point-to-point connection, the performance of UDP degrades very quickly as the robot moves away from the access point. This is in contrast to the Simple ACK and MCTP protocols which work very well out to a reasonable distance. Particularly, we see the rate for the confirmation protocols start at 1.0 and slowly drop to 0.85, at 18 meters. At this point the performance of the confirmation protocols begin to noticeably degrade with a precipitous drop from 0.85 to less than 0.1 in 4 meters.

3.3.2 Full System Validation

In our final experiment we incorporate our communication protocol, MCTP, with robust routing for a full system validation. This experiment seeks to show that by adding our communication

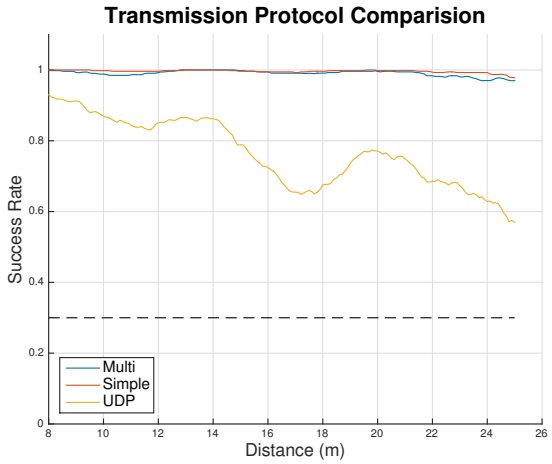
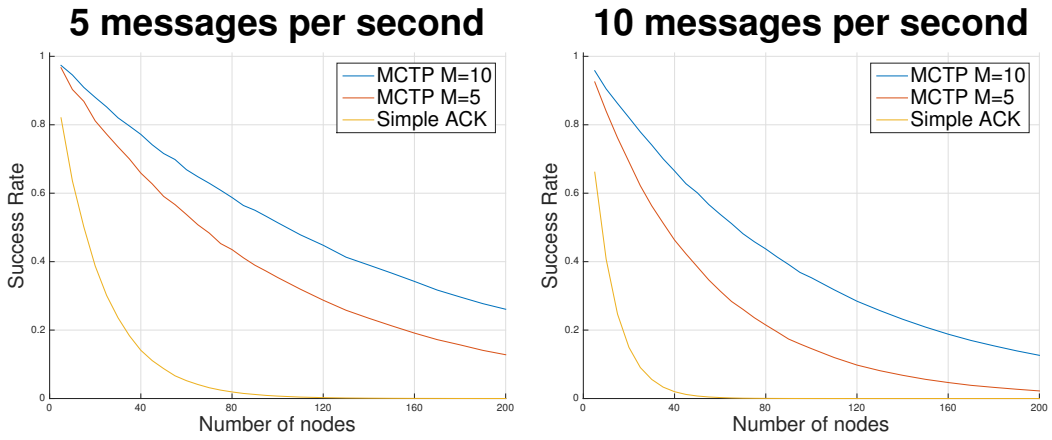


Figure 3.6: Plot for a team of 4 robots moving out while following the same paths as in Fig. 3.3.

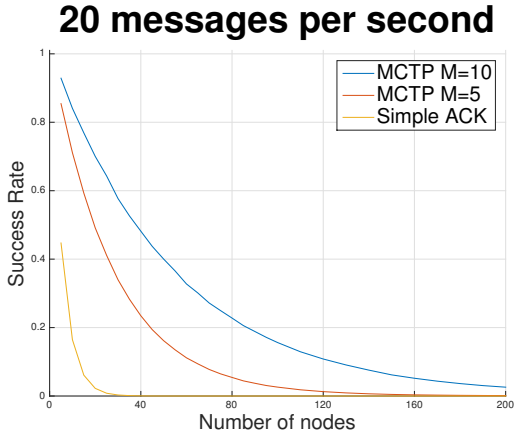
protocol to the robust routing solutions we can achieve near loss-less end-to-end communication between the sensing robot and the access point, even when direct communication is not possible. For this experiment the same parameters and paths are used as in the Levine-GRW experiment in Section 3.1.1, with only the communication protocol changing. The parameters used for the confirmation protocols are the same as those used in the experiments in Section 3.3.1.

When we incorporate the MCTP protocol the results immediately show improvement. As it can be seen in Fig. 3.6, when a confirmation protocol is used, the success rates increase dramatically. Using the Simple ACK and the MCTP protocols we see almost loss-less communication, even beyond 24 meters. The key result of this experiment is that the MCTP protocol, which is only sending confirmation messages for every 5 packets, has approximately the same reliability as the Single ACK protocol, with less than 0.025 maximum deviation between the two. By using MCTP with these parameters, compared to the Simple ACK, we can allow up to 5 times as many robots on the same confirmation channel. This means that for every robot added to the team, only 1.2 effective channels are required compared to 2 channels for Simple ACK.

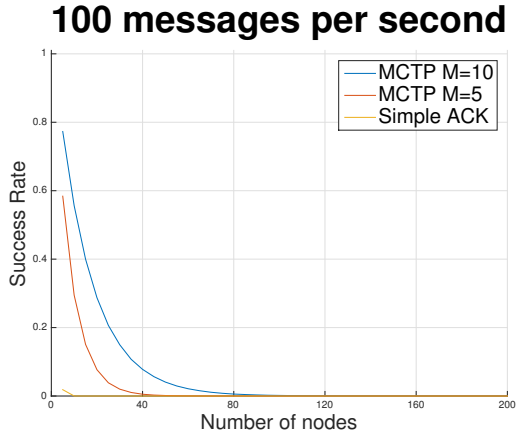


(a) Performance of MCTP compared to Simple ACK for $\mu = 0.2$

(b) Performance of MCTP compared to Simple ACK for $\mu = 0.1$



(c) Performance of MCTP compared to Simple ACK for $\mu = 0.05$



(d) Performance of MCTP compared to Simple ACK for $\mu = 0.01$

Figure 3.7: Simulated evaluation of MCTP for large-scale systems. Notice how MCTP outperforms a Simple-ACK protocol as the number of robots grows or the data rates increases.

3.3.3 Large-Scale Simulation

The performance advantage of MCTP relative to a simple ACK system is expected to become more marked as we increase the number of robots in the system. Due to the limited availability of robots we perform simulations to quantify these advantage. In the simulations we have created a network of robots transmitting data over point-to-point links using the MCTP and simple ACK protocols for confirmation. In these simulations each robot generates a packet of data every τ seconds, where τ is drawn from an exponential distribution with mean μ. The destination for

each packet is uniformly distributed over the other robots in the network. This allows us to vary both the number of robots in the network, as well as the average data input rate for each robot. These simulations assume that every data packet is successfully received and only the confirmation packets can be lost due to collision.

The results are shown in Figs. 3.7a-3.7d, which plots the average success rate of confirmation packets transmission as a function of the number of robots in the system, for a given value of μ. Since this is the average success rate the closer the value is to 1 the better the system is performing. The main item to notice is the wide gap in performance between Simple ACK and MCTP across all combinations of μ and the number of robots, except for $\mu = 0.01$ which completely saturates the network and effectively no confirmation is possible when the team is larger than 25. The resulting increase in the success rate is directly attributable to fewer confirmation packets being sent, since a single MCTP message contains much more information than a Simple ACK message.

As a conclusion the MCTP protocol outperforms a Simple ACK system throughout our simulations. Specifically, as the number of robots increase the drop in performance is much more gradual for MCTP compared to Simple ACK, and the same relationship is seen when the average input data increases. This highlights the benefit of MCTP as the number of robots in the team grows or the input data rates increase.

3.4 Summary

In this chapter we develop a lightweight communication protocol that provides reliable communication over ad-hoc wireless networks. We begin by examining the differences between robust and non-robust routing solutions when used to route data through an ad-hoc network created by mobile robots. From this examination we determine that the robust formulation is preferable to the non-robust since the reduction in rate variance in significant while the reduction in throughput is minimal. Next, we explore the possibility of utilizing one of the traditional communication

protocols, namely UDP and TCP, with the realization that neither are sufficient. This leads to the development of MCTP, which is a lightweight communication protocol designed specifically for use with robust routing solutions and low-cost low-power transceivers. We then highlight through experiments MCTP's ability to provide the same QoS, with less ancillary communication, when compared to traditional communication protocols. Next, we demonstrate the ability of MCTP to provide near loss-less communication for a sensing robot back to an access point through an ad-hoc wireless network. We conclude with simulations detailing the benefits of MCTP as the team size grows. The graceful degradation of MCTP is obvious when compared to a traditional protocol.

Chapter 4

Hybrid System

The task of real-time situational awareness requires that the robots be able to communicate reliably over the point-to-point communication links and efficiently move through the environment so as to maintain the communication network with minimal global coordination. As we have seen in Chapter 3, by leveraging robust routing solutions and MCTP it is possible to have reliable communication over an ad-hoc network of low-cost transceivers. While this was demonstrated in Section 3.3.2, the system used to control the motion and compute the routes was entirely centralized. As discussed in Section 2.2, a centralized architecture allows for globally optimal solutions but results in a decrease in efficiency, to the point of failure, as the number of robots on the team increases. In contrast to this, the purely distributed architecture is an approach that is able to scale appropriately as the team size increases but is unable to guarantee global optimality.

Since neither the centralized nor distributed architectures provide all the necessary properties we desire, the development of a hybrid system architecture is required. This architecture should be not only be capable of guaranteeing global optimality, but scales effectively to allow large teams. To achieve this, we isolate the portion of the real-time situational awareness problem that requires global coordination from the portion that does not. In doing this, we formulate two separate by coupled problems, one solved in a centralized manner and the other in a distributed manner. With

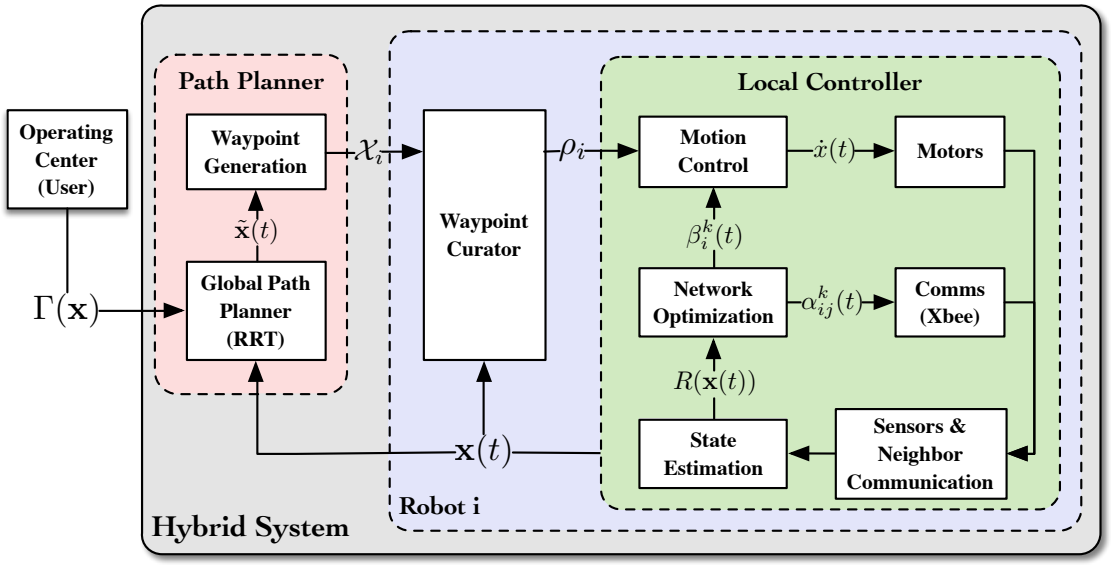


Figure 4.1: Hybrid architecture diagram. The red indicates the outer, centralized, loop of the system while the green indicates the inner, local controller, loop.

the formulation of two subproblems, we are able to minimize that amount of global coordination required, thus allowing the system to operate mostly in a distributed manner. The solution to the subproblems draw inspiration from previously developed systems, [24, 98], one centralized, the other distributed. Using these solutions, we systematically construct the hybrid architecture that is able to satisfy the requirements of real-time situational awareness for any size team. This resulting system is compared to existing systems through experiments demonstrating the benefits of a hybrid system architecture, in realistic situational awareness tasks.

4.1 Hybrid System Architecture

The goal of the hybrid system is to drive an arbitrary number of mobile robots through a complex environment while maintaining a minimum QoS in order to complete a given task. To accomplish this, we propose an architecture that consists of a two stage feedback system shown in Fig. 4.1. This architecture is composed of an outer centralized planning feedback loop and an inner distributed control feedback loop. The process is initiated by the user providing a global task function $\Gamma(\mathbf{x})$

to the outer loop. This begins the planning process which generates a set of dense *candidate* trajectories for the system that we denote as $\tilde{\mathbf{x}}(t) = \{\tilde{x}_i(t)\}_{i=1}^{N}$. Given the channel model that has been provided as input to the outer loop, these trajectories give an approximate robust solution to (2.1.9) where the network integrity constraints are given by (2.1.12). The candidate trajectory $\tilde{\mathbf{x}}(t)$ is never executed but rather fed to a waypoint generator that converts the dense trajectories into a series of waypoints for each robot,

$$\mathcal{X}_i = \left\{\tilde{x}_i(\tau_w)\right\}_{w=1}^{W}. \tag{4.1.1}$$

The waypoints in (4.1.1) are sampled at the same set of times $\{\tau_w\}_{w=1}^{W}$ for all robots and represent a decomposition of (2.1.9) into subproblems that can be solved by the distributed control inner loop.

The waypoints in (4.1.1) serve as sequential inputs to the distributed control loop. In contrast to the centralized loop which only operates when a new task is given, the distributed loop operates continuously on each robot. The distributed controller accepts a target location $x_{i,g}$ as input and attempts to drive the robot to that location while avoiding physical obstacles and preserving network integrity. The process that is used to implement this driving is distributed in that it relies on communication between adjacent robots only. When robot i receives a new set of waypoints $\mathcal{X}_i$ from the global planner its waypoint curator is responsible for updating the target location $x_{i,g}$. This is done by setting $x_{i,g}$ to the first waypoint in the series, i.e., by making $x_{i,g} = \tilde{x}_i(\tau_1)$. Then, when the distance to the target location falls bellow a given tolerance $\omega > 0$, namely, when $\|x_i(t) - x_{i,g}\| \leq \omega$, the waypoint is declared reached and $x_{i,g}$ is updated to the next waypoint in the series. The curator advances though successive waypoints until the final waypoint is reached at which time the trajectory is declared accomplished for robot i.

Notice that the candidate trajectory $\tilde{\mathbf{x}}(t)$ generated by the centralized planner is optimal for the model that is available. However, given the possibility for model mismatch, the trajectory

is not necessarily optimal during actual deployment. The distributed controller corrects for this mismatch because, due to the small communication overhead of its implementation, it can adapt to the conditions observed during execution. Thus, the hybrid system proposed here resolves the lack of adaptability of the centralized planner while avoiding the local minima that can limit the progress of the distributed control loop. We describe the centralized and distributed loops in the following sections.

4.1.1 Centralized Path Planning

The purpose of global path planning is to find a trajectory that solves the robust version of (2.1.9). A robust formulation of the network integrity constraints is used due to desire to construct long term trajectories that visit points in space for which the channel rates have yet to be measured. As we have seen in Section 2.1.2 there is more than one method to produce robust routing solutions. For this system we use the robust formulation in (2.1.12),

$$\begin{aligned} \boldsymbol{\alpha}(t) = \underset{\boldsymbol{\alpha}(t), a_\Delta}{\operatorname{argmax}} \quad & a_\Delta \\ \text{s.t.} \quad & \mathrm{P}\left[a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k + a_\Delta \right] > 1 - \epsilon \\ & \sum_{j,k} \alpha_{ij}^k(t) \leq 1. \end{aligned}$$

We assume that the variance estimate of the channels is available, thus we use this formulation as opposed to (2.1.11). To use the probabilistic formulation a model for the rates must be selected. For this system the model used for the rates is a Gaussian random variable. This implies that $R(x_i(t), x_j(t))$ can be completely characterized by its mean $\bar{R}_{ij}$ and variance $\tilde{R}_{ij}$. As we saw previously, when modeling the rates as random variables, the rates in (2.1.7) become random variables as well. Observe that the rate $a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$ has a normal distribution because the rates $R(x_i(t), x_j(t))$ are assumed to be Gaussian and $a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))$ is a linear function of $R(x_i(t), x_j(t))$.

As it follows from (2.1.7), the mean $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) := \mathbb{E}\left[a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))\right]$ of this Gaussian variable can be written as

$$\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) = \sum_{j=1}^{N} \alpha_{ij}^k(t)\bar{R}(x_i(t), x_j(t)) - \sum_{j=1, i\notin\mathcal{D}_k}^{N} \alpha_{ji}^k(t)\bar{R}(x_j(t), x_i(t)), \tag{4.1.2}$$

and the corresponding variance $\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) := \text{var}\left[a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))\right]$ is given by the expression

$$\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) = \sum_{j=1}^{N} (\alpha_{ij}^k(t))^2\tilde{R}(x_i(t), x_j(t)) + \sum_{j=1, i\notin\mathcal{D}_k}^{N} (\alpha_{ji}^k(t))^2\tilde{R}(x_j(t), x_i(t)). \tag{4.1.3}$$

Using the mean and variances in (4.1.2) and (4.1.3) and letting $\Phi^{-1}(\epsilon)$ stand for the inverse Gaussian complementary cumulative distribution function, we can write the probability constraint in (2.1.12) as

$$\frac{\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) - a_{i,m}^k}{\sqrt{\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))}} \geq \Phi^{-1}(\epsilon). \tag{4.1.4}$$

The constraint in (4.1.4), being dependent on the probabilistic model variables $\bar{R}(x_i(t), x_j(t))$ and $\tilde{R}(x_i(t), x_j(t))$, can be evaluated by the global path planner. We therefore use (2.1.9) with the probabilistic constraints in (2.1.12) to write the optimization problem

$$\begin{aligned}
&\min_{\dot{\mathbf{x}}(t), \boldsymbol{\alpha}(t)} \Gamma\left(\mathbf{x}\left(t_f\right)\right) \qquad (4.1.5)\\
&\text{s.t.} \quad \mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} \dot{\mathbf{x}}(s)ds, \quad \mathbf{x}(t) \in \mathcal{F},\\
&\qquad \bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k + \Phi^{-1}(\epsilon)\sqrt{\tilde{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t))},\\
&\qquad \sum_{j,k} \alpha_{ij}^k(t) \leq 1,
\end{aligned}$$

where, as in (2.1.9), the constraints hold for all terminals i, flows k, and times $t \in [t_0, t_f]$.

The formulation in (4.1.5) is first explored in [24] where the constraint in (4.1.4) is shown to

define a second order cone as long as $\epsilon < 0.5$ – which is not restrictive since we want ϵ to be small. Therefore, the determination of routing variables $\boldsymbol{\alpha}$ that satisfy this constraint can be written as a second order cone program if the formation $\mathbf{x}(t)$ is given. This implies that determining routing variables for a given formation can be done in polynomial time by using convex programing techniques [46]. In particular, checking if routing variables that satisfy the constraint in (4.1.4) exist is tractable, which in turn implies that finding formations that are feasible for the problem in (4.1.5) is also tractable. This is exploited in the solution of (4.1.5) with a Rapidly Exploring Random Tree (RRT) [42] as we explain in the next section.

Do notice that acquiring an accurate probabilistic model of reliabilities is itself challenging. The values of $\bar{R}(x_i(t), x_j(t))$ and $\tilde{R}(x_i(t), x_j(t))$ are dependent on shadowing and fading effects that can vary substantially in different propagation environments. The problem formulation in (4.1.5) circumvents this problem with the use of the robust routing constraint in (2.1.12). If the available propagation model is rough, this is captured in large values for the variances $\tilde{R}(x_i(t), x_j(t))$, which in turn result make it difficult to find formations that satisfy (4.1.4). This leads to conservative plans that can later be refined by the distributed controller which, different from the global planner, can rely on online modification of the propagation model.

Rapidly exploring random tree

The robust routing constraints in (4.1.5) modify the configuration free space $\mathcal{F}$. On top of physical obstacles and collision avoidance, we also need to remove formations for which satisfying (4.1.4) is not possible – which, as we argued before, can be done in polynomial time. We explore the resulting free space with a RRT. The RRT algorithm is initialized by first setting the current valid formation as the root of a tree. Then the following process is repeated until a formation that satisfies the task objective is added to the tree, $\Gamma(\mathbf{x}) = \Gamma(\mathbf{x}^*)$. A random point from the configuration space, corresponding to a formation, is drawn. The node nearest in the tree to this point is then found. For this configuration space a simple Euclidean distance is used to determine

the nearest node. Now using the nearest node as a starting point the system computes a path to the sampled point under the motion dynamics of the platform. The path is then truncated, if necessary, to a maximum length, called the steer distance, and divided into a series of intermediate formations. These formation are sequentially validated against (4.1.4) and verified to be collision free with respect to other robots and obstacles in the environment, [44, 50]. If there is a formation that violates (4.1.4) or results in a collision, the previous formation, if not the nearest node, is added to the tree with a branch from the nearest node. If all of the intermediate formations are valid and collision free the final formation in the truncated path is added to the tree with a branch from the nearest node. This process is repeated until a formation that satisfies the task objective is added to the tree. To expedite the searching process, the candidate formations are drawn from a probability distribution that is weighted along the shortest path between the current formation and the desired goal formation. This allows for a more efficient search for the majority of tasks.

The path through the tree starting at the current formation to the goal formation is then extracted. Since a node can only be added to the tree if the flow constraints are satisfied it is guaranteed that for every node in the final path the flow constraints are satisfied. This path corresponds to a feasible trajectory for each robot from its current location to a final location, $\tilde{\mathbf{x}}(t)$.

4.1.2 Distributed Controller

The purpose of the distributed controller is to manage the mobility and network routing of an individual robot using the waypoints generated by the centralized controller [cf. (4.1.1)]. This dual mandate requires that we solve both the motion control and the network routing. To accomplish this, we run concurrently a continuous-time motion-gradient control and a discrete-time dynamic computation of optimal communication variables [98].

For the motion-control portion of the distributed controller we employ a navigation function that is capable of driving the robot to a goal location $x_{i,g}$ while avoiding obstacles [68,69]. However,

obstacles here are not physical but determined by the need to guarantee network integrity. Assume then that routing variables $\boldsymbol{\alpha}(t)$ are given and recall that network integrity is defined as the satisfaction of the QoS requirements in (2.1.8). If we further introduce a strictly positive tolerance $e > 0$ we can thus define the obstacle function for robot i associated with the k^{th} constraint as

$$\beta_i^k(\mathbf{x}(t)) \triangleq \sum_{j=1}^{N} \alpha_{ij}^k(t)\bar{R}(x_i(t), x_j(t)) - \sum_{j=1}^{N} \alpha_{ji}^k(t)\bar{R}(x_j(t), x_i(t)) - a_{i,m}^k + e. \tag{4.1.6}$$

The function $\beta_i^k(\mathbf{x}(t))$ is positive when the k^{th} QoS requirement for robot i is satisfied within the tolerance e for the current formation $\mathbf{x}(t)$, and negative otherwise. This allows a gradient controller to treat the zero points of $\beta_i^k(\mathbf{x}(t))$ as the border of a virtual obstacle that, if crossed, would result in a violation of the integrity of the k^{th} flow. Observe that, different from the centralized controller, this QoS constraint can be accurately evaluated at the current location because the propagation model can be adapted to observations. Also notice that the tolerance e simply implies a reduction of the minimum acceptable rate from $a_{i,m}^k$ to $a_{i,m}^k - e$. They are kept separate to emphasize that the distributed controller requires some leeway to increase its range of motion for a given set of communication variables.

The obstacle defined by the function $\beta_i^k(\mathbf{x}(t))$ in (4.1.6) is associated with robot i and flow k. For robot i all the QoS obstacle functions can be combined into the single network integrity obstacle function,

$$\beta_i(\mathbf{x}(t)) = \min_{k=1,\ldots,K} \beta_i^k(\mathbf{x}(t)). \tag{4.1.7}$$

Integrity of all flows at robot i is guaranteed within the tolerance e if the joint obstacle function is $\beta_i(\mathbf{x}(t)) > 0$. To create an attraction to $x_{i,g}$ we use the goal potential function $\rho_i(\mathbf{x}(t)) = \|x_i(t) - x_{i,g}\|^2$. Using this definition of $\rho_i(\mathbf{x}(t))$ and the obstacle function in (4.1.7) we can define the navigation function,

$$\phi_i(\mathbf{x}(t)) = \frac{\rho_i(\mathbf{x}(t))}{\left(\rho_i(\mathbf{x}(t))^\kappa + \beta_i(\mathbf{x}(t))^2\right)^{1/\kappa}}, \tag{4.1.8}$$

where the order parameter satisfies $\kappa > 2$ and has to be chosen sufficiently large. This navigation function has the desirable properties of being $\phi_i(\mathbf{x}(t)) \in [0,1]$ always, satisfying $\phi_i(\mathbf{x}(t)) = 0$ when $\mathbf{x}(t) = x_{i,g}$, and being such that $\phi_i(\mathbf{x}(t)) \to 1$ when a QoS requirement is about to be violated. Taking advantage of these properties we can drive robot i towards $x_{i,g}$ while guaranteeing network integrity with the gradient descent controller

$$\dot{x}_i(t) = -\nabla_{x_i}\phi_i(\mathbf{x}(t)). \tag{4.1.9}$$

The value of κ is used to control the regions that are affected by the obstacles, the larger κ is the more localized the effects are to the obstacles. As shown in [68,69], the controller in (4.1.9) is able to reach $x_{i,g}$ while avoiding obstacles that are not intersecting and spherical. The obstacle defined by (4.1.7) is not spherical and it may be that (4.1.9) stops at a local optimum. This is not a concern because the sampling of waypoints is done fine enough to preclude this possibility. Notice that in the complex environments considered here it is also necessary to avoid physical obstacles. This is standard problem that we can solve, e.g., with a modification of (4.1.7) to include the distance to these physical obstacles.

Assuming that feasible routing variables $\boldsymbol{\alpha}(t)$ satisfying $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k$ are available for all formations for which these variables exist, the controller in (4.1.9) coupled with proper generation of waypoints would drive the team to a formation that solves (2.1.9). What is left, therefore, is the design of a distributed mechanism to find these feasible routing variables. We do so in the following section.

Adaptation of routing variables

The motion control of the robot is predicated on the virtual obstacles created in (4.1.6) which are computed directly from the routing solution $\boldsymbol{\alpha}(t)$. When starting at a waypoint and moving to the next, we have available the routing solution $\boldsymbol{\alpha}(t)$ that has been computed by the centralized

controller. This solution can be used for initialization, but an accurate description of the obstacle space necessitates the routing solutions $\boldsymbol{\alpha}(t)$ used in the controller in (4.1.9) to adapt as the robots move. In order to adapt these variables without requiring knowledge of the variance we adopt the approach taken in (2.1.11) to determine robust solutions.

Specifically, extract the network integrity constraint $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,m}^k$ from (2.1.11) and rewrite it as $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) = a_i^k \geq a_{i,m}^k$. The idea here is to adapt the routing variables so that the expected rates a_i^k are as large as possible – but not smaller than the minimum requirement $a_{i,m}^k$. To do so introduce weights $w_i^k > 0$ and $w_{ij}^k > 0$ and define the weighted proportional fair utility $U_i^k(a_i^k) = w_i^k \log(a_i^k)$ as well as the weighted quadratic penalty terms $V_{ij}^k(\alpha_{ij}^k) = -w_{ij}^k(\alpha_{ij}^k)^2$ that we incorporate into the optimization problem

$$\begin{aligned} \boldsymbol{\alpha}(t) = & \underset{a_i^k, \alpha_{ij}^k}{\operatorname{argmax}} \sum_{k=i}^{K} \sum_{i=1}^{N} \left[U_i^k(a_i^k) + \sum_{j=1}^{N} V_{ij}^k(\alpha_{ij}^k) \right] \\ & \text{s. t.} \qquad \bar{a}_i^k(\boldsymbol{\alpha}, \mathbf{x}(t)) = a_i^k \geq a_{i,m}^k, \quad \sum_{j,k} \alpha_{ij}^k(t) \leq 1. \end{aligned} \tag{4.1.10}$$

Some remarks are in order. To guarantee that a solution to (2.1.9) is found we need to find, for any given spatial formation $\mathbf{x}(t)$, a set of routing variables that satisfy $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) = a_i^k \geq a_{i,m}^k$ for all robots i and flows k. However, there are, in general, many variables that satisfy these constraints. The formulation in (4.1.10) resolves this indeterminacy by selecting the variables $\boldsymbol{\alpha}(t)$ that maximize the objective $\sum_{k=i}^{K} \sum_{i=1}^{N} \left[U_i^k(a_i^k) + \sum_{j=1}^{N} V_{ij}^k(\alpha_{ij}^k) \right]$. Since these variables are optimal in (4.1.10) they are feasible in particular, but the presence of the fair utility term $U_i^k(a_i^k) = w_i^k \log(a_i^k)$ also makes the difference between the estimated achieved rate $\bar{a}_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) = a_i^k$ and the minimum rate $a_{i,m}^k$ large. Assuming that rates $R(x_i(t), x_j(t))$ change slowly in space, this allows more freedom of movement for fixed routing variables and, consequently, less frequent recomputation of the solution of (4.1.10). The quadratic penalty terms $V_{ij}^k(\alpha_{ij}^k) = -w_{ij}^k(\alpha_{ij}^k)^2$ hedges the solution against errors in the estimation of the rates $R(x_i(t), x_j(t))$ because they ensure that a link is not overly utilized when similar links are available.

The problem formulation in (4.1.10) answers the question of which routing variables to plug in the definition of the obstacle function in (4.1.6) but, as formulated, (4.1.10) requires global coordination to compute the optimal routing solution. A distributed method to solve (4.1.10) follows from the observation that, for a given spatial formation $\mathbf{x}(t)$, the problem is convex and can therefore be equivalently solved in the dual domain with a gradient descent method. Introduce then a non-negative dual variables $\lambda_i^k(t_n)$ associated with each of the $a_i^k(\boldsymbol{\alpha}, \mathbf{x}) = a_i^k$ constraints in (4.1.10), where t_n is used to track the current iteration. These variables can be grouped into a matrix, $\boldsymbol{\lambda}(t_n) \in \mathbf{R}^{N \times K}$. Using the dual variables and the constraints we can write the Lagrangian,

$$\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x}) = \\ \sum_{k=i}^{K} \sum_{i=1}^{N} \left[U_i^k(a_i^k) + \sum_{j=1}^{N} V_{ij}^k(\alpha_{ij}^k) + \lambda_i^k \left(\sum_{j=1}^{N} \alpha_{ij} \bar{R}(x_i, x_j) - \sum_{j=1,\, i \notin \mathcal{D}}^{N} \alpha_{ji} \bar{R}(x_j, x_i) - a_i^k \right) \right]. \quad (4.1.11)$$

We can rearrange the terms in (4.1.11) into a sum of local Lagrangians, $\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x}) = \sum_{i=1}^{N} \mathcal{L}_i(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x})$, where

$$\mathcal{L}_i(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x}) = \sum_{k=1}^{K} U_i^k(a_i^k) - \lambda_i^k a_i^k + \sum_{j=1}^{N} \left[V_{ij}^k(\alpha_{ij}^k) + \alpha_{ij}^k \bar{R}(x_i(t), x_j(t))(\lambda_i^k - \lambda_j^k) \right]. \quad (4.1.12)$$

Notice that $\mathcal{L}_i(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x})$ only depends on robot i's information, a_i^k, λ_i^k, and α_{ij}^k, as well as only the λ_j^k's for which $R(x_i(t), x_j(t)) > 0$. This indicates that in order to compute the value of $\mathcal{L}_i(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x})$ robot i is only required to collect the λ_j^k of its immediate neighbors. This can be achieved by a simple exchange of λ_i^k between all neighboring pairs. Upon receipt of its neighbors' variables $\lambda_j^k(t_n)$ robot i is able to compute its optimal rates and its part of the routing solution, at time t_n,

by solving,

$$a_i^k(t_n), \left\{\alpha_{ij}^k(t_n)\right\}_{j=1}^N = \text{argmax } \mathcal{L}_i(\boldsymbol{\lambda}(t_n), \boldsymbol{\alpha}(t_n), \mathbf{x}(t_n)).$$
$$\text{s.t.} \qquad a_i^k \geq a_{i,m}^k, \ \sum_{j,k} \alpha_{ij}^k(t) \leq 1. \tag{4.1.13}$$

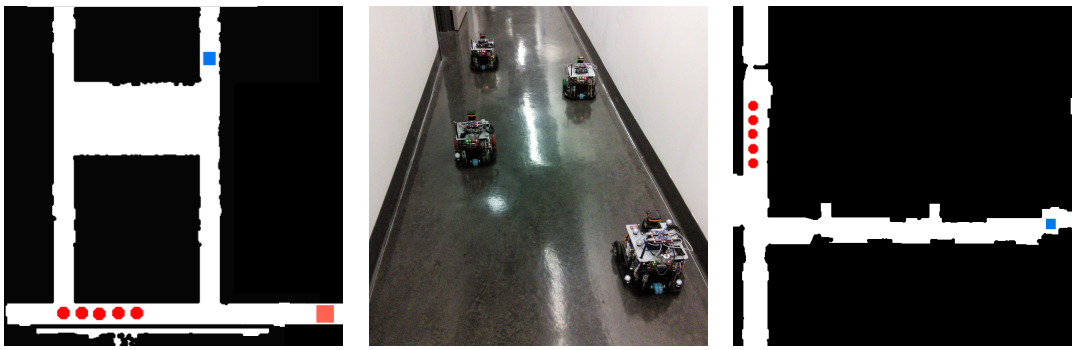
After the optimal rates and routes are determined for time t_n the next step is to update the value of λ_i^k. To maintain the non-negative requirement for λ_i^k, we use a non-negative projection $\mathbb{P}[y]$, which returns y is $y \geq 0$ and 0 if $y < 0$. Using this projection we update $\lambda_i^k(t_n)$ by following $\nabla_{\lambda_i^k} \mathcal{L}_i(\boldsymbol{\lambda}, \boldsymbol{\alpha}, \mathbf{x})$, using the values of $a_i^k(t_n)$ and $\alpha_{ij}^k(t_n)$ found in (4.1.13),

$$\lambda_i^k(t_{n+1}) = \mathbb{P}\left[\lambda_i^k(t_n) - \epsilon\Big(\sum_{j=1}^N \alpha_{ij}^k(t_n)\bar{R}(x_i(t), x_j(t)) - \sum_{j=1}^N \alpha_{ji}^k(t_n)\bar{R}(x_j(t), x_i(t)) - a_i^k(t_n)\Big)\right], \tag{4.1.14}$$

These updated values are then shared with all the robots within communication range so they can be used in the next iteration of (4.1.13). This process is repeated and converges to the optimal routing solution when the formation is static. If the formation is changing the resulting solutions will be near optimal, and the deviation from optimality is dependant on the frequency of the iterations and the allowable velocity of the robots.

4.2 Additional Environment

In this chapter, two distinct environments are used. The first is Levine-GRW, Fig. 4.2a, and the second is the Towne building, Fig. 4.2c, both at the University of Pennsylvania. These two environments were chosen due to their different RF characteristics. These differences are derived from the construction date and materials used in the two buildings. The Levine building was built in 1996 and consists of mostly metal framing and drywall, while the Towne building was built in early 1900's and consists of mostly brick and concrete. These two environments allow for a better test suite for the hybrid system, as opposed to Levine vs. Levine-GRW. An image of the robots



(a) Operating region in Levine-GRW.

(b) Image of the robots in Levine-GRW.

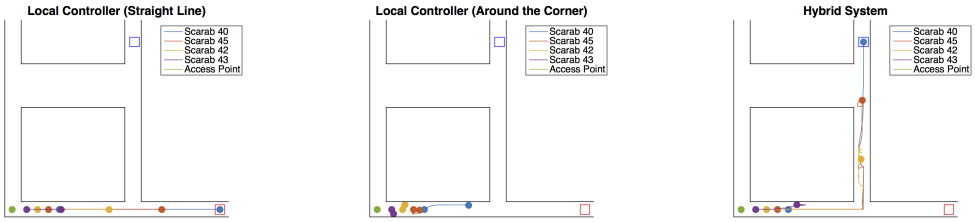
(c) Operating region in Towne.

Figure 4.2: Environments used in simulations and experiments

operating in the Levine-GRW environment can be seen in Fig. 4.2b. Due to large differences in the RF environments and to demonstrate the flexibility of the hybrid system to mismatched channel models, we are using a function that is a polynomial fitting of experimental curves found in the literature [2] for the local controller channel model.

4.3 Simulations

In this section, we highlight the benefits of our hybrid approach over a distributed system, while retaining the benefits of such a system. In the first set of simulations, a team of 4 mobile robots and 1 access point are given the task of moving one specific robot to a goal location in a complex environment. Two goal locations are used and it is shown that purely distributed operation fails while the hybrid system can successfully reach the goal. In the second simulation a large team is tasked with supporting one robot moving through a complex environment. This simulation demonstrates the ability of the hybrid system to scale with the number of robots in the team. While the physical communication layer is not simulated in these scenarios, the systems are operating as they would during a deployment; rates are estimated, dual variables are exchanged, routes are computed and motion is constrained based on the underlying network obstacles. A set of experiments with the full system, including the physical communication layer, are presented in



(a) Waypoint is straight ahead, no obstacles. Local controller is able to achieve the goal.

(b) Waypoint is around a corner. Local controller fails to achieve the goal.

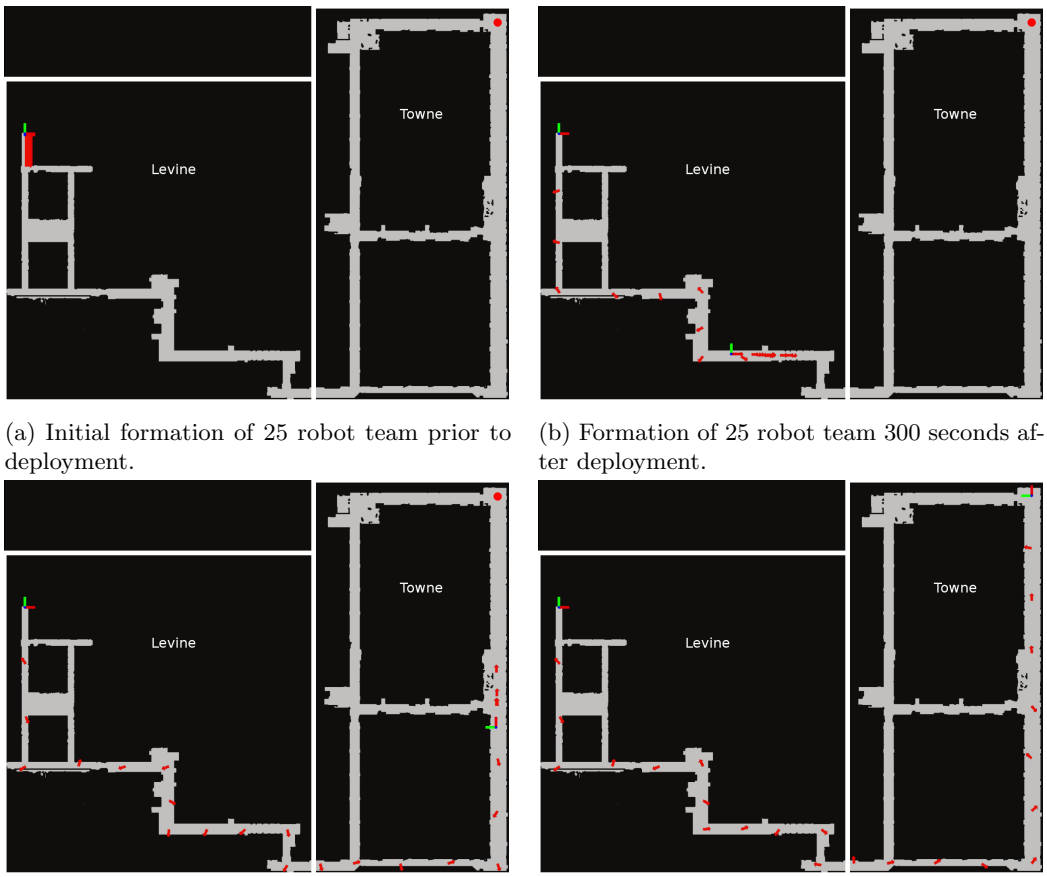
(c) Waypoint is around a corner. Hybrid system is able to achieve the goal.

Figure 4.3: Simulation results for local and hybrid systems. For all tests the goal location is 19 meters away.

Section 4.4.

4.3.1 Local Minima

In this set of simulations, we demonstrate the limitations of a purely local controller. Using only the controller described in Section 4.1.2, the team of 4 robots and an access point are given the task of driving *Scarab*40 to a specific goal location. For all three simulations in this section, the goal is 19 meters away from the access point, only the location of the goal is changed. The first location given was straight along the lower hallway in the Levine-GRW map, Fig. 4.2a, as indicated by the red square. The second was around the lower right corner in the same building, which is indicated by the blue square in the Fig. 4.2a. The resulting trajectories for all three simulations are plotted in Figs. 4.3a-4.3c. In Fig. 4.3a, it can be seen that the robots successfully assemble into a formation that allows the sensing robot to reach the goal, as indicated by the final position of *Scarab*40 being inside the red square. In contrast, Fig. 4.3b shows that the local controller alone is not capable of driving the team into a valid formation when the goal is around the corner. This is due to the local minima that is created by the attractive force of the goal being cancelled out by the repulsive force from the wall and the attractive force from network preservation. The final simulation in this section shows the performance of the hybrid system when given the same task of turning the corner. As seen in Fig. 4.3c the team is able to successfully turn the corner



(a) Initial formation of 25 robot team prior to deployment.

(b) Formation of 25 robot team 300 seconds after deployment.

(c) Formation of 25 robot team 600 seconds after deployment.

(d) Formation of 25 robot team 900 seconds after deployment.

Figure 4.4: Evolution of a 25 robot team that is supporting one robot, indicated by red and green axis, from the initial starting formation in the upper left corner to the goal location in the upper right corner, indicated by the red circle.

and assemble into a formation that allows the sensing robot to reach the goal, as indicated by *Scarab*40 reaching the blue square. This is achieved because each robot is given a series of 3 goals locations that change the location of the local minima and thus allow the team to reach a valid final formation.

4.3.2 Large Scale Deployment

Another scenario that we explore in simulation focused on the ability of the hybrid system to operate in a complex environment when the team size is large, $N = 25$. The environment used in

this scenario is one floor of the Levine and Towne buildings, shown in Fig. 4.4a, which has over 850 m^2 of floor space. The access point, $i = 25$, and the sensing robot, $i = 24$, are indicated by the thick red and green axes, while the remaining 23 support robots, $i = \{1, \ldots, 23\}$ are indicated by red arrows. The team begins in a formation $\mathbf{x}(t_0)$ located in the upper left corner of the Levine building. It is tasked with supporting a single QoS requirement with $a^1_{24,m} = 0.3$, $a^1_{j,m} = 0.0$ for all $j \neq 24$, and $\mathcal{D}_1 = 25$, while robot 24 moves to the goal location, $x_{24,g}$, in the upper right corner of the Towne building. In this environment the shortest path from $x_{24}(t_0)$ to $x_{24,g}$ is over 200 m. Upon receipt of $x_{24,g}$ the global planner determines trajectories for each robot, which are then passed to the waypoint generator and converted into waypoints for the local controllers.

Remark 4.3.1. Due to the size of the environment and the number of robots, the global planner restricted samples for the RRT to points that were within a meter of robot 24's shortest path. While this restriction limits the set of possible final formations, it allows the system to find feasible trajectories more quickly, as long as there are sufficient number of robots on the team.

With the waypoints from the global planner, the local controllers begin executing their trajectories. Snapshots of the team's formation in the environment at 0, 300, 600, and 900 seconds are shown in Figs. 4.4a - 4.4d. As shown in the figures, the team is able to successfully deploy into a formation that allows robot 24 to successfully traverse the environment and reach $x_{24,g}$. In this deployment every robot is critical to the data path from robot 24 to the access point due to the complexity of the environment. Note, since each robot is critical to the network, each robot has sufficient back haul to support robot 24's data back to the access point. Therefore, given the problem formulation in (4.1.10) any location, $\hat{x}_{24}$, where $R(\hat{x}_{24}, x_j) \geq a^1_{24,m}$, is a location at which robot 24 can collect data. With this understanding we see that robot 24 is able to retrace its path back to the access point and network integrity will be maintained for the duration of its travel. Since this environment is more complex than the environment in Section 4.3.1, it can be safely assumed that even with knowledge of their final location, the local controllers would would not be able to successfully reach those locations, due to local minima. This highlights that not only is

it necessary to find a final formation that supports $x_{24,g}$ but intermediate waypoints are needed to ensure proper avoidance of local minima. This simulation was run on a 2.7 Gigahertz Intel i7 laptop with 16 Gigabytes of RAM to demonstrate the lightweight nature of the local controller. After the waypoints were determined, all 25 local controllers ran in parallel in real-time.

4.4 Experimental Evaluation

Our work is motivated by the uncertainty and difficulty in modeling real-world wireless communication. Since our primary objective is to maintain a reliable wireless communication network, it is important that we evaluate the system under the realistic RF conditions. As noted previously, for these experiments we used the *Scarab* platform, [48], with XBee transceivers, see Section 2.3.1.

The first set of experiments we ran, Section 4.4.1, compared the hybrid system to the full system developed by Fink et al. in [24]. This system consists of two centralized parts, the path planner and the motion controller. The path planner is the same as the one used in our hybrid system. The motion controller executes the plans determined by the path planner in a synchronous closed loop manner. This means that each robot is given a location to drive to and then wait till given the next location. The next location is not published until all of the robots have reached their goal. This is implemented in order to preserve the guarantee that at each intermediate formation network integrity is preserved. While this approach does provide more control over the evolution of the formation and underlying wireless network, it is rigid and susceptible to breakage. An example of a scenario that would cause such a breakage is shown in Section 4.4.2. In that set of experiments one of the support robots incurs a temporary motor failure in-between two formations and the results causes the centralized system to lose network integrity, while the hybrid system preserves network integrity.

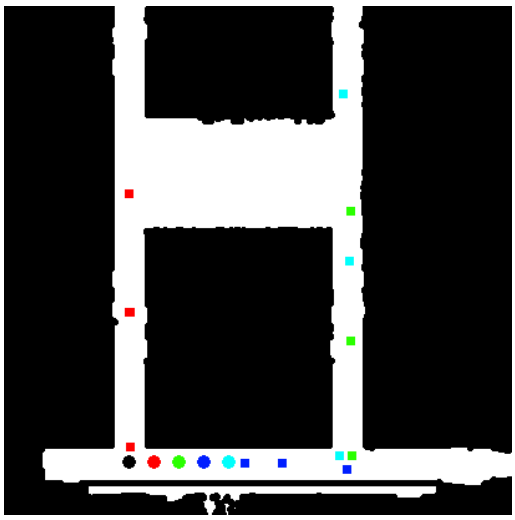
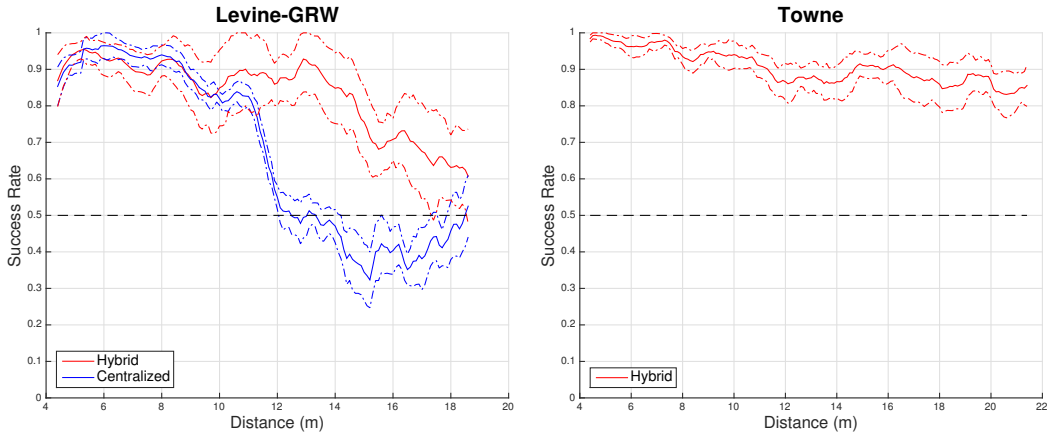


Figure 4.5: The waypoints used in Section 4.4.1. Robots are color coded with the initial formation indicated by the circles and the waypoint as squares.

4.4.1 System Comparison

In the initial set of experiments, we compare the successful packet transmission of our hybrid system to the centralized system developed by Fink et al. There were three sets of experiments run for this section. Each set consisted of ten trials, with only one data flow, $a^1_{4,m} = 0.5$. The first two sets provide the comparison between the hybrid and centralized systems in Levine-GRW, while the third highlights the performance of the hybrid system in a different environment, the Towne building. For the first two sets, the centralized planner was used to find the trajectories that allowed the team to complete the goal, which was reach the blue square from the initial formation shown in Fig. 4.2a. With these trajectories the waypoint generator was used to reduce the number of waypoints to three as shown in Fig. 4.5. These sets of waypoints were then used by both the centralized motion controller and the local controllers, to remove any bias incurred by different input waypoints. The results of the ten trials are plotted in Fig. 4.6a, where the solid line represents the average over all the trials and the dotted envelope shows the one σ bounds. There are a few items to note; first, there is a portion of the data in which the average success rate for the centralized system falls below 0.5; this is mostly due to a mismatch between the channel model



(a) Experimental results for centralized and hybrid systems in Levine-GRW. The solid line is the average performance and the dashed colored lines are +/- 1 σ bounds. The black dashed line is the minimum input data rate for the lead robot.

(b) Experimental results for the hybrid system in the Towne building. The solid line is the average performance and the dashed line is the +/- 1 σ bounds. The red dashed line is the minimum input data rate for the lead robot.

Figure 4.6: Experimental results for Levine-GRW and the Towne building.

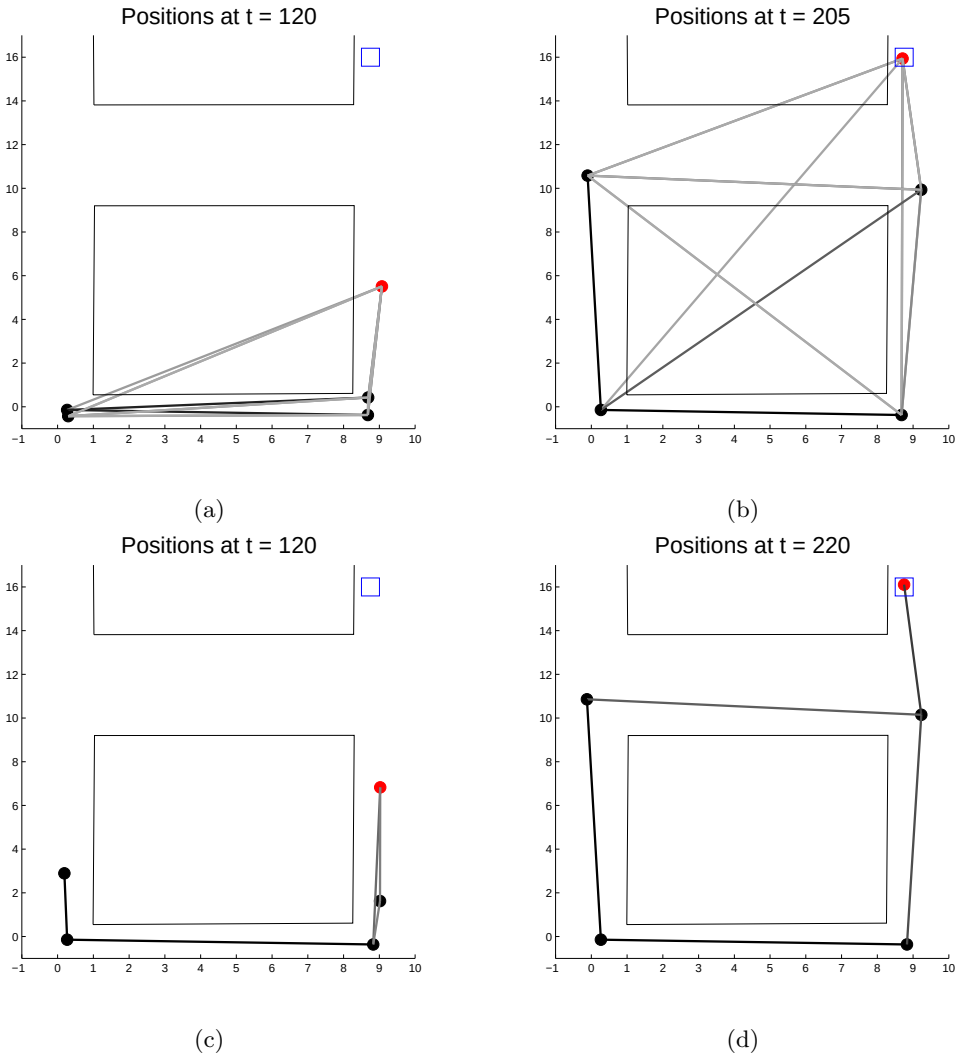
and the actual environment. The second item to notice is how well the hybrid system performs. Even the one σ bound stays above the required data rate. This is mostly due to the robots locally optimizing their trajectory and not moving in straight lines. Another item to note is the spread on the one σ bounds. Since the centralized system is including an estimate of the channel variance the spread is much less than the hybrid system which is only a proxy for the channel variance. Also, since the hybrid system allows for deviations to locally optimize, the trajectories taken by the robots is not always the same compared to the tightly controlled trajectories executed by the centralized system. The final item to note is the divergence of the results for the two systems at 12 meters. While the hybrid system continues to exceed the required data rates, the centralized system drops off dramatically to marginally meeting the requirements. The reason for this is at 12 meters the sensing robot turns the corner and must rely on the support robots to relay data back to the access point. Since the centralized system is planning for future unknown links rates it adopts a conservative approach with respect to a single link. This conservative approach is useful when planning but it does not leverage the current state of the environment and team formation.

In contrast the local controller in the hybrid system is constantly optimizing for performance based on the environment and team's formation. Therefore, it can achieve a higher level of performance when compared to the centralized systems due to better utilization of current information. An example of this is seen in Fig. 4.7, where the location and routing probabilities are plotted for one trial of the experiment. For both systems, two snapshots in time are taken, $t = 120$ and at the completion of the task. In the first time instance, the formations are not identical. This is due to the local deviations performed by the hybrid system, but the final formations match.

In the third set of experiments for this section the same task, drive around a corner, was completed but in the Towne building shown in Fig. 4.2c. Again, the hybrid system was given the blue square as a goal location for the sensing robot, and the initial formation is indicated by the red circles. Ten experiments were run with $a^1_{1,m} = 0.5$, and the results are plotted in Fig. 4.6b. The system performs remarkably well, with the one σ bounds well above the desired results. This is most likely due to the Towne building having wider hallways compared to Levine-GRW and therefore the amount of multi-path interference being reduced when the robots are in the center of the hallway. Also, the same model parameters were used as in Levine-GRW; thus the superior performance could indicate that the channel model is conservative with respect to the RF environment in Towne when compared to Levine-GRW.

4.4.2 Dynamic Response

In this section of tests, we highlight a major benefits of using a local controller, as opposed a centralized waypoint system, namely dynamic response to unexpected events. In these experiments, as with those in the previous section, the goal was to drive around the corner in Levine-GRW to a goal location, but during deployment one of the robots has a temporary restriction to its motion. A temporary restriction in motion could be caused by events such as an actual failure of the physical motor or an obstacle or person blocking the path of the robot. Similar to the previous section, feasible trajectories were found and passed to the waypoint generator. The resulting way-



(a) (b) (c) (d)

Figure 4.7: These figures show a series of formations and the resulting routing probabilities experienced during the experiments in Section 4.4.1. Figs. (a) and (b) correspond to the centralized system experiments and Figs. (c) and (d) correspond to the hybrid system experiments. The darkness of the lines connecting the robots indicate the routing probabilities used for that link.

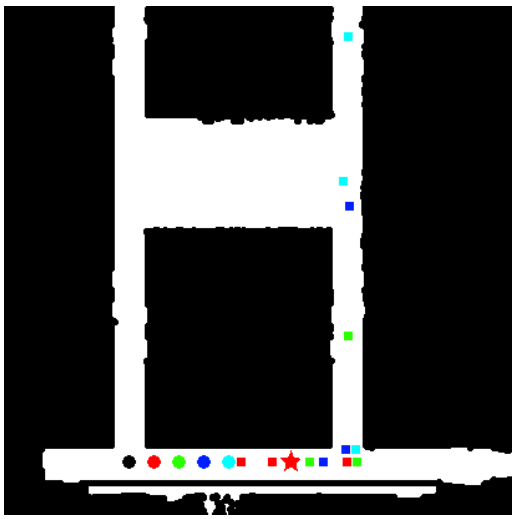


Figure 4.8: The waypoints used in Section 4.4.2. Robots are color coded with the initial formation indicated by the circles and the waypoint as squares. The red star indicates the location at which the support robot suffers the motor failure.

points are shown in Fig. 4.8, as in the previous section each robot has three waypoints. This set of experiments were run just as the previous section was but when *Scarab*43 reaches the red star in Fig. 4.8 its motor is disabled for 120 seconds, to simulate a temporary restriction in motion. In Figs. 4.9a and 4.9c we plot the team's formation for the centralized and hybrid systems during the stall period, and in Figs. 4.9b and 4.9d we plot the formations at the completion of the experiment. In these plots the sensing robot is a red circle, the support robots are black circles, the final team formation is shown as blue squares, and *Scarab*43 is highlighted by a red square. Notice that since *Scarab*43 stalls after the second set of waypoints in Fig. 4.8 the centralized system attempts to reach the final formation. This is seen in Fig. 4.9a by all the robot except *Scarab*43 reaching their goal location. After *Scarab*43 recovers from the stall it moves to it's final location and the team is in the correct final formation. This does not occur when the hybrid system is used due to the team dynamically reacting to the stall and preventing the sensing robot from advancing farther. As shown in Fig. 4.9c by the red circle not reaching its blue square.

To analyze the network performance of these tests we ran two more experiments where *Scarab*43 does not stall using the same configuration as the prior tests. The results of the two experiments

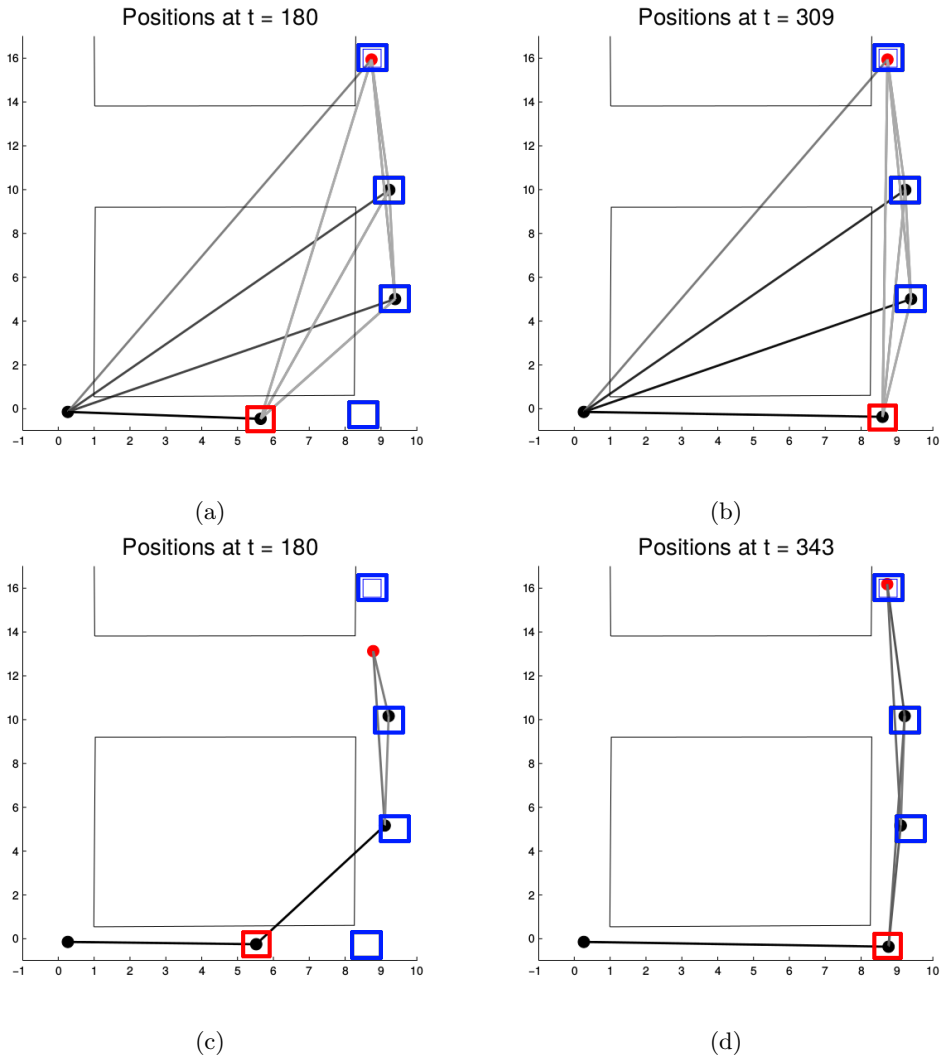
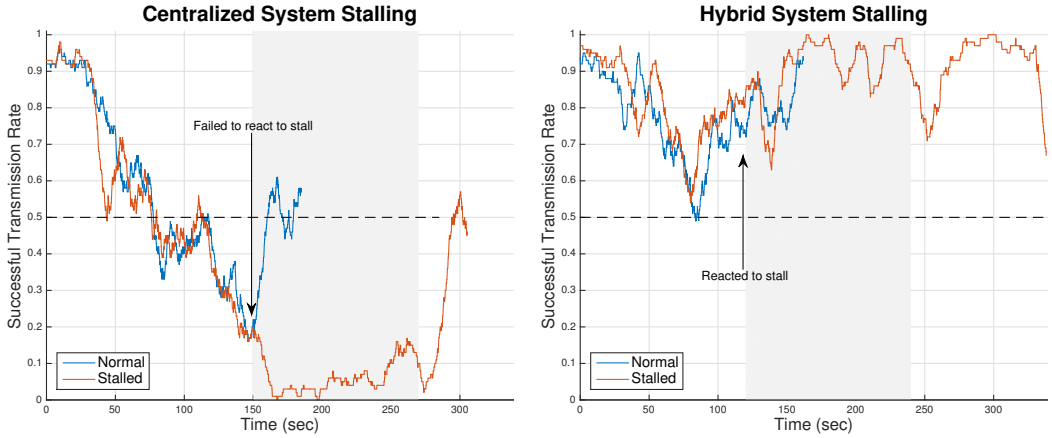


Figure 4.9: These figures show a series of formations and the resulting routing probabilities experienced during the experiments in Section 4.4.2. Figs. (a) and (b) correspond to the centralized system experiments and Figs. (c) and (d) correspond to the hybrid system experiments. Figs. (a) and (c) show a snapshot of the formation when *Scarab*43 has stalled. The darkness of the lines connecting the robots indicate the routing probabilities used for that link.



(a) The centralized system fails to adjust to the motor failure and the network suffers greatly.

(b) The hybrid system is able to adjust the motion of the robots to overcome the motor failure.

Figure 4.10: Experimental results highlighting the hybrid systems ability to dynamically adjust to motor failures. In both figures two separate experiments are plotted. The blue line is from an experiment under normal conditions and the red line is from an experiment where there is a motor failure. The shaded region indicates the time the motor failed for the stalled experiment.

for the centralized and hybrid systems are plotted in Figs. 4.10a and 4.10b. In these plots the red and blue lines are the data rate of system with and without the stall, which is indicated by the shaded region. It can be seen that prior to the stall the two lines are in agreement for both systems as is expected since there has not been an unexpected event yet. When the stall occurs we see that the two lines in Fig. 4.10a diverge, while the they do not in Fig. 4.10b. The divergence in Fig. 4.10a is due to the formation deviating greatly from the one that was verified by the centralized planner. After the stall is recovered from we see that the network performance returns to the desired value. In contrast in Fig. 4.10b we see that the network performance never suffers from the robots being out of position. This is because when the stall occurs the other members of the team react accordingly, specifically the sensing robot halting its motion. These experiments show how the hybrid system is more robust to dynamic changes in the environment and other obstacles that may arise during the execution of a task when compared to the more brittle waypoint synchronization of the centralized approach.

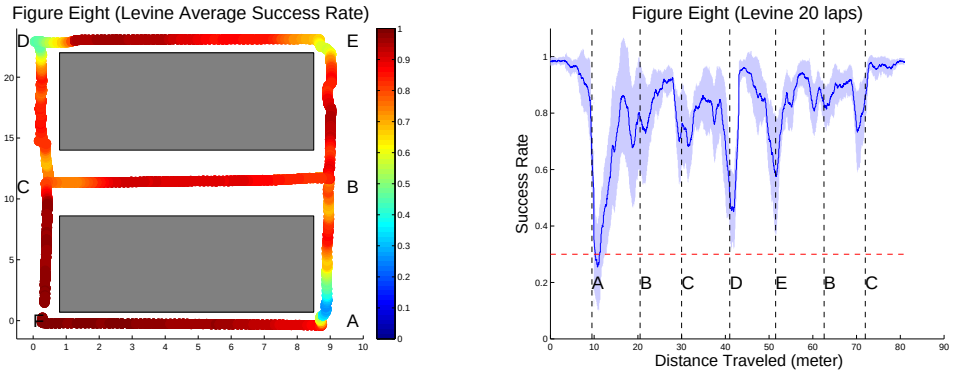
4.5 Situational Awareness Task

The previous sections demonstrated, through simulations and experiments, that the hybrid system is able to control the motion of the team so that the sensing robot is able to reach a specific location. In this section, we demonstrate that by building upon this ability, we can extend the system to complete complex tasks with minimal user input. One such task is long duration monitoring or patrolling a series of hallways. For this task, the sensing robot is not moving to a specific location, but instead the requirement is to visit multiple sensing locations, all the time maintaining the desired QoS.

We begin by decomposing the task of patrolling a hallway into a series of operations. First, the system determines a path for the patrol robot that visits all the sensing locations and returns to its current location to create a loop. This loop allows for repeated execution of the generated path without compromising the QoS. Next, the global planner uses this path to determine a goal formation for the support robots that maintains the QoS for the majority, if not entirety, of the patrolling robot's motion. This goal formation, including the first sensing location, is then used as the desired formation for the RRT in global planner. With this desired formation the system operates just as it does in the single location scenario. After finding the trajectories and disseminating the waypoints, the local controllers drive the robot to their goal locations. Upon reaching their goals, the robots are able to adjust their location to optimize the communication network in response to the rest of the team. This allows the team to react to locations along the patrolling robot's path that are not supported by the goal formation, but are still feasible for patrolling.

4.5.1 Non-confirmation Protocol

In the first experiment we wish to quantify the ability of the hybrid system to provide real-time situational awareness as the sensing robot patrols the hallway. To provide an accurate measure



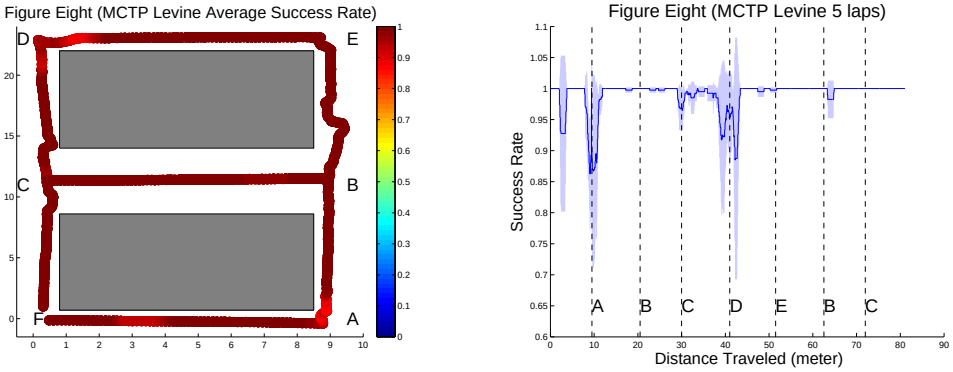
(a) Heat map for the experiments run in Section 4.5.1.

(b) The average and 1 σ bounds for the the experiments run in Section 4.5.1.

Figure 4.11: Experimental results from patrol task using UDP.

of only the hybrid system's performance we use UDP, as in our tests in Section 3.1. For this experiment we use a team of 3 robots supporting a patrol robot as it moves through a figure eight hallway. The location of this experiment is Levine-GRW and the desired QoS is set to $a^1_{4,m} = 0.3$. The team of 4 robots and an access point begin in the lower left corner near location F in Fig. 4.11a with sensing location (A, B, C, D, E, B, C). The global planner uses this order of sensing locations to determine an optimal formation for the support robots. The resulting formation covers the entire path by placing the support robots at locations B, C, and D. With the path covered, every location along the patrol robot's path will have sufficient network connectivity to support the required QoS. Thus, the local controllers are not required to deviate from the formation. In this experiment the robot executes the figure eight path a total of 20 times.

The resulting data rates for each lap are overlaid in Fig. 4.11. In Fig. 4.11a we plot the average data rate, signified by the color, at each location along the path. In Fig. 4.11b, we plot the average and one σ bounds as a function of distance traveled. The vertical dotted line indicate the waypoints. As with the previous experiments, even the one σ bound is above the required rate, $a^1_{4,m} = 0.3$, for the majority of the experiment. Note that other than right after location A, the system maintains the required QoS. This drop off is consistent across laps, as evidenced by the σ bounds not spreading out. We attribute this result to the delay in the convergence of the



(a) Heat map for the experiments run in section 4.5.2.

(b) The average and 1 σ bounds for the the experiments run in section 4.5.2.

Figure 4.12: Experimental results from patrol task using MCTP.

routing algorithm to the new optimal solution. This is due to the dramatic change in the solution from a direct path to the access point to a multi-hop path through two support robots.

4.5.2 Confirmation Protocol

In the final set of experiments the hybrid system was combined with the Multi-Confirmation Transmission Protocol (MCTP) from Chapter 3. Incorporating a transmission protocol that supports confirmations, such as MCTP, allows the hybrid system to further mitigate the random fluctuations in the wireless links. Another benefit of using MCTP, as previously detailed, is that it was designed specifically for the wireless ad-hoc networks created by the hybrid system. These two facts allow the team to achieve even more reliable end-to-end communication. An example of this is seen when the patrolling experiment from the previous section is repeated but with MCTP, instead of UDP. The results from these experiments, in which five laps instead of the twenty were executed, are shown in Fig. 4.12. Again a heat map showing the average success rate is plotted in Fig. 4.12a and the average success rate as a function of distance traveled with a one σ bound is plotted in Fig. 4.12b. The benefit of incorporating a confirmation protocol is immediately apparent. The system maintains an average success rate greater than 0.85, even though the desired rate was only 0.3. This is achieved without changing any of the system requirements, therefore

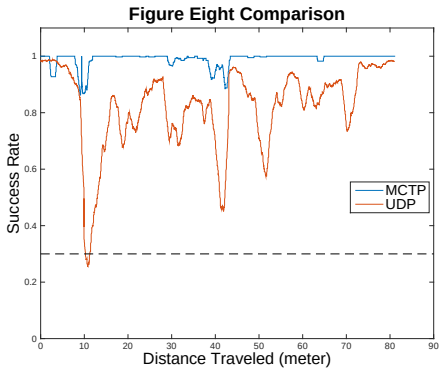


Figure 4.13: Average success rate as a function of distance for the the figure eight experiments in Sections 4.5.1 and 4.5.2.

demonstrating that even with lower input requirements the system will optimize the end-to-end data rate. This property allows the system to provide a minimum guarantee for the performance of the network, while not sacrificing the possibility of more optimal operation. A more direct comparison between the UDP and MCTP experiments is shown in Fig. 4.13. Notice that there are similar drops in the performance between the two, but the magnitude of the drops is much less in the MCTP experiments, indicating a much more reliable end-to-end transmission.

This experiment also highlights the value of the hybrid system's design. Since the focus of the hybrid system is on motion control and network routing, it resides entirely in the network layer of the OSI networking model. This allows the hybrid system to be incorporated into any larger system that requires a networking layer that is also capable of motion control. As demonstrated in the previous experiments, the hybrid system can operate independently, or as demonstrated in this experiment, it can be incorporated with other OSI compliant components to construct a more complex system. This provides a system-level flexibility to incorporate advancements in other layers of the OSI model without requiring modification to the hybrid system.

4.6 Summary

In this chapter, we develop a hybrid system that is able to drive a team of robots through a complex environment while preserving network integrity. The system is composed of two main subsystems arranged in a feedback loop. The outer loop is responsible for generating trajectories for the team and the inner loop in responsible for using those trajectories as a roadmap to successfully complete the given task. This construction allows the team to complete the situational awareness task in a distributed manner while avoid local minima.

We demonstrate the abilities of the system extensively through simulations and experiments. In simulation, we highlight a specific task that a distributed controller is unable to complete, but is trivial for the hybrid system. Additionally, we verify the ability of the system to operate without a reduction in performance, even when the team consists of 25 robots. Through experiments, we compare the performance of the hybrid system to a centralized system and show the hybrid system's ability to optimize the robot's trajectories to achieve a higher end-to-end data rate with less coordination overhead. Then, we emphasize the dynamic nature of the hybrid system in experiments where one of the support robots has a temporary motor failure that results in the centralized system losing network integrity, but the hybrid system adapts and maintains the network.

We conclude with an application that is well suited to the hybrid system, patrolling a series of hallways. In this application a patrol robot must repeatedly visit a series of sensing locations while transmitting data back to an access point. The hybrid system is able to position the support robots such that there is never a break in the end-to-end link. We perform this task two times; first, using UDP to demonstrate the ability to maintain a minimum end-to-end link, and then using MCTP to demonstrate a near loss-less end-to-end link.

Chapter 5

Simultaneous Communication-Aware Localization and Mapping

The ability of a team of robots to move through a known environment while providing real-time situational awareness over a wireless ad-hoc network with minimal global coordination, as demonstrated in Chapter 4, is a major step towards completing our objective. While this an improvement, there still remains one capability that is required for such teams to be useful in realistic scenarios. That capability is operation in unknown environments. As mentioned previously, the term unknown environment could be used to describe a variety of locations, such as a building that has experienced a partial collapse, a recent construction, or simply, a building that has not been mapped prior. In this chapter, we present a system that is capable of efficiently constructing a meaningful representation of the environment while supporting real-time transmission of timely sensor data back to a central location. Therefore, we say this system is preforming the task of Simultaneous Communication-Aware Localization and Mapping (SCLAM).

5.1 Unknown Environments

The existence of an accurate map of the environment is critical to the abilities of the hybrid system demonstrated in Chapter 4. The map is used throughout the system, specifically in the localization system to estimate the position of the robot, in the channel estimation system to identify if the link between two robots is obstructed by an obstacle, and in the global path planner to determine collision free trajectories. The impact of an inaccurate, or incomplete map, can be dramatic to each of these systems.

5.1.1 Occupancy Grid

Before we can begin to quantify the impact of dynamic maps on the components, we must understand their properties. This system is designed to operate with a probabilistic representation of the environment. Specifically, a discrete occupancy grid, $\mathbf{m}$, that divides the space into K independent planar cells m_k. Cells take values $m_k \in \{0, 1\}$ with $m_k = 1$ signifying the presence of an obstacle in the corresponding region of space, and 0 signifying the space is free. The occupancy grid is represented probabilistically by the map random variable M, composed of individual cell random variables M_k [84]. These cell variables are modeled as Bernoulli with $p(m_k) := \mathrm{P}\left(M_k = 1\right)$ denoting the probability of m_k containing an obstacle. Individual cells are further assumed independent of each other, which implies that the probability distribution of the map is completely characterized by the occupancy probabilities of individual cells. We therefore have that the probability distribution $p_M(\mathbf{m}) = p(\mathbf{m})$ of the map M is given by

$$p(\mathbf{m}) = \mathrm{P}\big(M = \mathbf{m}\big) = \prod_{m_k=1} p(m_k) \prod_{m_k=0} [1 - p(m_k)]. \tag{5.1.1}$$

We remark that we use the map belief in (5.1.1) for discussions and probability computations but that in practice the map representation is simply given by the collection $\{p(m_k)\}_{k=1}^{K}$ of cell occupancy probabilities. Additionally, we define $\hat{\mathbf{m}}(t)$ as the maximum likelihood estimate of the

map given the probability distribution at time t. This is done by thresholding the probability distribution $p(\mathbf{m})$. Specifically, we set $\hat{m}_k(t) = 0$ when $p(m_k) \leq \nu_{free}$, $\hat{m}_k(t) = 1$ when $p(m_k) \geq \nu_{obstacle}$, and $\hat{m}_k(t) = -1$ otherwise.

5.1.2 Component Modifications

The components in the original system operate with a map similar to $\hat{\mathbf{m}}(t)$, except that due to the environment being known only the values 0 and 1 are present. Therefore, given that the maximum likelihood map, $\hat{\mathbf{m}}(t)$, is likely to contain unknown and unexplored cells, those cells where $\hat{m}_k(t) = -1$, the components that rely on a map must be augmented.

Localization

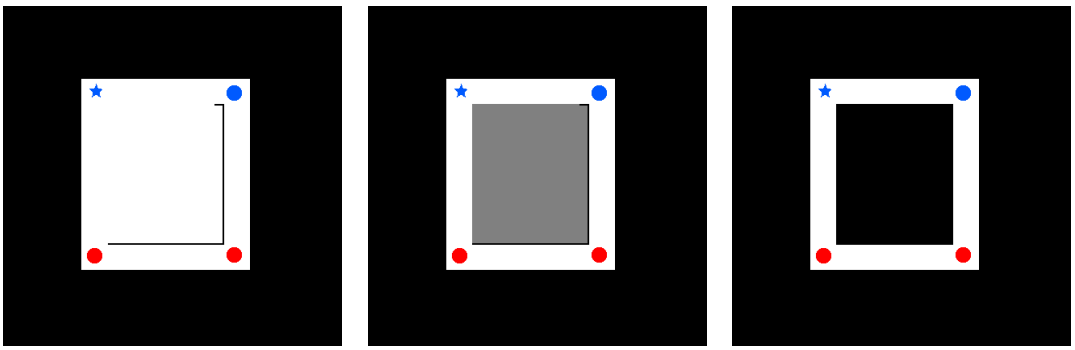
We begin with the localization system used by the robots. The process of localization involves the robot obtaining sensor measurements, $\mathbf{z}_t$, in our case the range returns from a 2-dimensional scan done by the Hokuyo. The range returns $\mathbf{z}_t$ and current estimate of the robots location, $\hat{x}_i(t)$ are used to create a point could, or collection of points in space that represent objects. This point cloud is then matched with the expected measurements given the estimated location. Through a series of iterations, $\hat{y}(t)$, an estimate of the laser's location given the measurements is produced. Then, $\hat{y}(t)$ is used to update $\hat{x}_i(t)$ to produce a new estimate of the robot's location. If both the map and laser are accurate the resulting localization errors, $\|\hat{y}(t) - \hat{x}_i(t)\|$, are small and the location of the robot can be known precisely. Due to the map in our situation containing many unknown locations, theses errors increase. To mitigate an increase in the errors, two modifications are made. First, the reliance on the local odometry, which is the robot's ability to estimate its motion without any external information, such as a map, is increased. This is possible due to the accuracy of the laser and the laser odometry system developed by the MRSL. The other step taken to mitigate the errors is the locations in which the robots can move to are limited to areas where the environment has been sufficiently mapped. While this limitation could prevent the robots

from moving into advantageous positions, the benefit of reduced localization errors far outweighs the possible gains.

Channel Estimation

The next component that must be modified is the channel estimation system. This system is used by both the global planner and local controller in the hybrid system. Therefore, limiting its errors is paramount. To provide an estimate of the channel rate, the system collects the indices of all the cells that the link between two robots passes through into a set, $\mathbf{c}$. When the environment is known, it is trivial to determine if there are any obstructions, $m_k = 1$ for some $k \in \mathbf{c}$, since all of the cells take a value of 0 or 1. With this information, it is able to properly model the expected channel rate to account for shadowing, but when the map is incomplete this capability is diminished. This is because there are now three possible outcomes: i) there are only open cells along the link ($\hat{m}_k(t) = 0$ for all $k \in \mathbf{c}$); ii) there is at least one cell with an obstacle ($\exists\ k \in \mathbf{c}$ s. t. $\hat{m}_k(t) = 1$); and iii) there are only open and unknown cells along the link ($\hat{m}_k(t) \in \{-1, 0\}$ for all $k \in \mathbf{c}$). The first two outcomes can be treated in the same manner as when there is full information. It is the third, and new outcome, that must be addressed. When $\hat{m}_k(t) \in \{-1, 0\}$ for all $k \in \mathbf{c}$ it is impossible for the system to know if there is an obstacle in any of the unknown cells. This requires that the system take a conservative approach and assume that any unknown cell contains an obstacle that will attenuate the signal. As with the limitations placed on localization, the confidence that the link will only be better than the estimate, and not worse, is preferred over the risk of violating network integrity.

A simple example of this can be seen when the current maximum likelihood map of the world is the one shown in Fig. 5.1b, and two extreme cases are shown in Figs. 5.1a and 5.1c. In this simple example, the robots are only able to communicate when the line of sight is unobstructed. In this example the black indicates an obstacle, white indicates open, and grey indicates unknown. The goal is to have the blue circle move to the blue star while using the two red circle, which are



(a) One possible state of the environment.

(b) Current maximum likelihood map of the environment.

(c) One possible state of the environment.

Figure 5.1: A simple example of why the channel estimation system must take a conservative approach to link estimation when the environment is unknown. If communication requires line-of-sight between robots, indicated by the circles, and the blue circle want to move to the star it must assume that the map in (c) is the real state of the world and not (a).

fixed access points, for communication. If the estimation system did not take the conservative approach and assumed that unknown cell did not contain obstacles, the blue robot would move towards the star expecting a link with the left circle. This would not violate network integrity if the map in Fig. 5.1a is the true state, but if the map in Fig. 5.1c is the true state then network integrity would be lost, and the correct trajectory is along the lower hallway.

The affect of this conservative approach is realized when computing the communication rate margin. To indicate that the channel rate estimates are the result of incomplete information, we replace $R(x_i(t), x_j(t))$ in (2.1.7) with $\hat{R}(x_i(t), x_j(t))$ to get,

$$\hat{a}_i^k(t) = \sum_{j=1}^{N} \alpha_{ij}^k(t)\hat{R}(x_i(t), x_j(t)) - \sum_{j=1, i\notin\mathcal{D}_k}^{N} \alpha_{ji}^k(t)\hat{R}(x_i(t), x_j(t)). \tag{5.1.2}$$

In (5.1.2), we use $\hat{a}_i^k(t)$ to denote that the resulting communication rate margin are also the result of incomplete information. Due to $\hat{R}(x_i(t), x_j(t))$ being the best estimate possible, the resulting $\hat{a}_i^k(t)$ must be used instead of $a_i^k(t)$. This results in conservative behavior in both the motion and routing solutions exhbited by the local controller. An important observation here is that as the map estimate $\hat{\mathbf{m}}(t)$ becomes closer to the actual map, the link rate estimates, $\hat{R}(x_i(t), x_j(t))$,

become closer to the actual rates. Thus, the flow rate estimates, $\hat{a}_i^k(t)$, become closer to actual data rates and the system returns to the scenario considered in Chapter 4.

Global Path Planner

The final component that relies on an accurate map is the global path planner. This component uses the map in two ways, one for channel estimation, which has already been discussed, and the other for physical path planning. In order to accommodate the lack of information, in regards to the motion planning, a similar approach to channel estimation is used. In that, unknown locations are treated as obstacles. As highlighted in the channel estimation discussion this conservative approach is required, but this time for the sake of safety.

The impact of this is noticed in the construction of the feasible configuration space. In order to safely maneuver the robots, we must modify the formulation in (2.1.5) to first take into account the physical constraints imposed on the robots by the current estimated map of the environment, $\hat{\mathbf{m}}(t)$. Therefore, we define $\mathcal{F}_{ro}(t)$ as the set of all cells that have a high probability of not being open, $\mathcal{F}_{ro}(t) = \{m_k : \hat{m}_k(t) \neq 0)\}$. Then, using the same construction of $\mathcal{F}_{rr}$, we define

$$\mathcal{F}(t) = \mathbb{R}^{2N} \setminus \mathcal{F}_{ro}(t) \setminus \mathcal{F}_{rr}, \tag{5.1.3}$$

and require that $\mathbf{x}(t) \in \mathcal{F}(t)$.

5.2 Autonomous SCLAM

With an understanding of how each component can mitigate the effects of dynamic maps, we turn our attention to how to complete a situational awareness task, given an unknown environment. For this chapter, we consider a team with the same composition as in Section 2.1, but with one addition. On this team, there is a single robot, $i = l$, that is capable of generating a map of the environment that can be shared with the other members of the team. We refer to this robot as the

lead robot. Since this robot is obtaining actionable information, in terms of situational awareness, all of the resources of the team are dedicated to supporting the flow of real-time date from this robot back to the access point, $a^1_{l,m} > 0$ and $a^1_{i,m} = 0$ for all other i. For this reason we referred to the other members of the team as support robots.

The ultimate goal of the system is to have the lead robot autonomously map the environment, as quickly as possible, while the support robots provide the communication network needed to relay data from the lead robot to the access point. The system must also adjust to changes in the communication requirements, caused by changes in the current estimate of the map, so as to minimize the delay experienced by the map updates at the access point. This ability is built upon the hybrid approach developed in Chapter 4, with the necessary modifications.

The first modification to the system is a result of the robots no longer being homogeneous, since one robot now has a capability the others do not. To provide this ability, the lead robot replaces its localization subsystem with a Simultaneous Localization and Mapping (SLAM) subsystem, which is now providing both an estimate of the robot's location, $\hat{x}_l(t)$ and a representation of the environment in a probabilistic sense, $p(\mathbf{m})$, as well as a maximum likelihood sense, $\hat{\mathbf{m}}(t)$.

With the introduction of autonomous mapping, the process of the user providing a task potential $\Gamma(\mathbf{x}(t))$ is removed. This is due to the expectation that the system should generate $\Gamma(\mathbf{x}(t))$ dynamically, based on the current understanding of the environment. Since the objective is to map the environment, which only relies on the location of the lead robot, $\Gamma(\mathbf{x}(t))$ is reduced to a function that relates the location of the lead robot to a goal location, $\Gamma(\mathbf{x}(t)) = \|x_l(t) - x_{l,g}\|^2$. This location, $x_{l,g}$, should be determined such that the map can be completed in as little time as possible. Thus, we employ an information theoretic approach to this process, so that the maximum information gain can be achieved as the lead robot moves. The optimal trajectory, which can be interpreted as a sequence of $x_{l,g}$'s, is found by an Information Theoretic Explorer, whose construction is presented in detail in Section 5.2.1.

This change to a dynamic determination of $\Gamma(\mathbf{x}(t))$ requires modifications to the operational

flow of the hybrid system. The hybrid system was constructed with the assumption that changes to the overall task were infrequent, but with dynamic generation of task this assumption is no longer valid. As such, we inverted the order of operation in the system so that there is more reliance on the local controller, and less on the global planner. Instead of relying on the global planner to determine a trajectory for each robot every time $\Gamma(\mathbf{x}(t))$ changes, the dynamic hybrid system initially uses the local controller on robots to drive to the goals. Therefore, the Information Theoretic Explorer interacts only with the local controller on the lead robot, passing the optimal trajectory to the local controller as a series of waypoints, $\mathcal{X}_l$, just as the global planner did in the original system. Thus, the Information Theoretic Explorer is located on the lead robot – although it can physically reside at some other computational element if that is desirable. This reliance on the local controller does introduce the possibility that a local minima will prevent the team from supporting the lead robot as it executes its trajectory. When this does happen, the team will remain trapped in the local minimum unless assistance is provided, which comes from the global planner when necessary. The global planner's role has therefore been reduced from being deeply embedded in the original system to now operating as a fail-safe, only used when the local controllers are unable to adequately solve the problem.

This resulting architecture is shown in Fig. 5.2. Consider the time $t = t_0$, at which time the Information Theoretic Explorer must generate an optimal trajectory for the lead robot. To accomplish this, it requests the current probabilistic interpretation of the environment, $p(\mathbf{m})$, and estimated location of the robot, $\hat{x}_l(t_0)$, from the SLAM component. Using this the Information Theoretic Explorer then determines an optimal trajectory, $\mathcal{X}_l$, over a finite time interval $t \in [t_0, t_0 + \tau]$. This trajectory when followed provides the robot with maximal information to refine the map. The trajectory, $\mathcal{X}_l$, is then passed directly into the local controller of the lead robot, which it attempts to follow. Concurrently, the lead robot is publishing the maximum likelihood representation of the environment, $\hat{\mathbf{m}}(t)$, every τ_p seconds. This results in the localization and channel estimation systems on the other robots operating with the most recent estimated map,

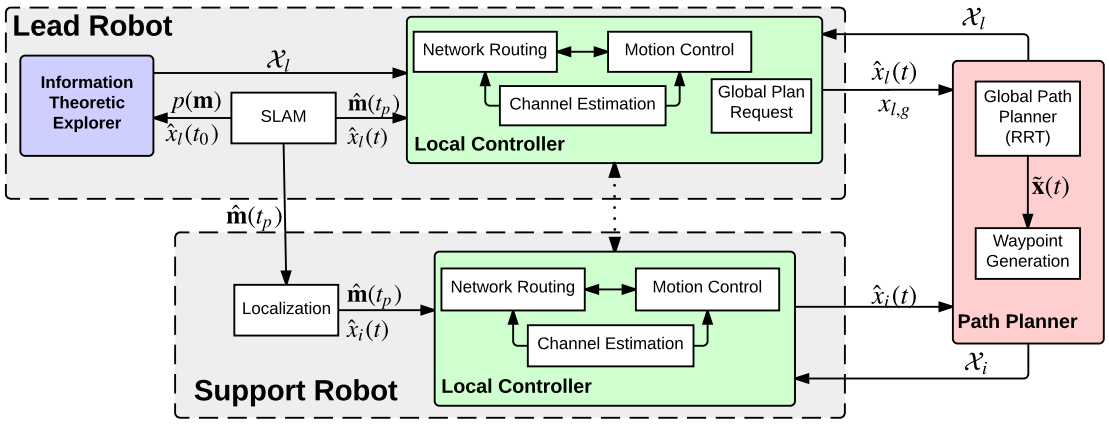


Figure 5.2: System architecture for SCLAM mobility and communication planning and control. The main components are an Information Theoretic Explorer that finds good trajectories for lead robot and a mobility and communication control algorithm that maintains a viable communication network. The Information Theoretic Explorer resides in the lead robot but communication control is distributed. A global path planner is used as a fail safe mechanism. Map estimates improve estimates of communication channel, which improve the quality of the communication network that supports the mapping and exploration task.

$\hat{\mathbf{m}}(t_p)$, where $t_p = n\tau_p$ and n is the largest integer such that $t_p < t$. While the lead robot is following the trajectory, the support robots are dynamically adjusting to the motion of the lead robot so as to provide the communication link back to the access point. This is done in the same manner as the original system, where the local controllers exchange the dual variables, and follow their navigation function, which is composed of physical and virtual communication obstacles. The main difference is that the rates and virtual obstacles are now constructed using the incomplete version of the communication margin, $\hat{a}_i^k(t)$. When the lead robot completes the trajectory prescribed by the Information Theoretic Explorer, the process is repeated with t_0 updated to the current time.

As noted, there is the possibility of the team being trapped in a local minima. To identify this, the leader monitors the amount of time it have been executing the given trajectory. When the execution time exceeds a time threshold T_l, where $T_l > \tau$, the team is halted and a global plan is requested. Upon reception of a global plan request, the path planner collects the current estimate of the formation $\hat{\mathbf{x}}(t)$, and map, $\hat{\mathbf{m}}(t)$, to determine the trajectory for the entire team, such that

the lead robot can reach the final location of the trajectory provided by the Information Theoretic Explorer. Only the final location is used since the intermediate points along the trajectory may not be feasible from a network integrity perspective. When the final trajectory is determined, it is sampled and disseminated in the same manner as Section 4.1.

Another modification that is made to the system pertains to the minimum end-to-end data rate requirement. In the previous experiments, the value of $a^1_{l,m}$ captured the rate necessary to sustain a consistent flow of data back to the access point. When performing SCLAM, there are two flows of data originating at the lead robot. The first is a consistent flow of data from a primary sensor, similar to the previous interpretation, and the second is a dynamic flow of newly obtained map data. Both of these flows are destined for the access point, but the primary sensor data is given priority, thus the system must adapt to minimize the delay experienced by the map updates when the bandwidth is insufficient. The method in which the system is augmented to address this is presented in Section 5.2.2.

These modifications and new architecture result in a system that is highly dynamic and operates almost entirely in a distributed manner.

5.2.1 Autonomous Exploration

In this section, we focus initially on the problem of a single robot autonomously mapping an environment. This problem can then be integrated with the dynamic hybrid system described in the previous section. We begin by noting that the map distribution $p(\mathbf{m})$, and an estimate $\hat{x}(t)$ of the current position of the robot, are updated with environmental observations z_t. The observation model of the sensors in the robot is given in the form of the probability distribution $p(z_t \,|\, x_t, \mathbf{m})$ that represents the probability of observing z_t when the position of the robot is $x_t = x(t)$ in the given environment $\mathbf{m}$. Notice that the environment is not known but rather described by the probability distribution $p(\mathbf{m})$. It is therefore of interest to recall the definition of the joint

probability distribution of the map and the observations which we write as

$$p\left(\mathbf{m}, z_t \,\middle|\, x_t\right) = p\left(z_t \,\middle|\, x_t, \mathbf{m}\right) p(\mathbf{m}), \tag{5.2.1}$$

as well as the marginal distribution of the observations z_t that we obtain by summing the joint distribution in (5.2.1) over all possible maps,

$$p\left(z_t \,\middle|\, x_t\right) = \sum_{\mathbf{m}} p\left(\mathbf{m}, z_t \,\middle|\, x_t\right) \tag{5.2.2}$$

The goal is to plan trajectories that map the environment efficiently, thus we require the robot to follow a path for which the future observations are maximally informative about the environment. These maximal information trajectories follow from information measures that we compute based on the probability distributions in (5.1.1)-(5.2.2).

Trajectories with maximum information gain

To measure the information gain of a trajectory for the robot, we rely on the Cauchy-Schwarz Quadratic Mutual Information (CSQMI) [61] between the map and the candidate trajectory. Begin by considering two arbitrary probability distributions $p_A(a)$ and $p_{A'}(a')$ with a *common* support $\mathcal{A}$. Interpreting these distributions as vectors in a Hilbert space, we can define their inner product as

$$\langle p_A(a),\ p_{A'}(a')\rangle = \int_{\mathcal{A}} p_A(a) p_{A'}(a)\, da \tag{5.2.3}$$

and their energies as $\|p_A(a)\|^2 = \langle p_A(a),\ p_A(a)\rangle$ and $\|p_{A'}(a')\|^2 = \langle p_{A'}(a'),\ p_{A'}(a')\rangle$. The Cauchy-Schwarz ratio,

$$\langle p_A(a),\ p_{A'}(a')\rangle^2 / \|p_A(a)\|^2 \|p_{A'}(a')\|^2,$$

is then a measure of how similar the distributions $p_A(a)$ and $p_{A'}(a')$ are to each other. Observe that if the random variable A is discrete, the integral in (5.2.3) is replaced by a sum.

Consider now a time $t = t_0$ at which the map belief is $p(\mathbf{m})$ and the current position of the robot is x_{t_0} and a time horizon τ. Between times $t = t_0$ and $t = t_0 + \tau$ the robot moves through a series of T equally spaced poses $\mathbf{x}_{t_0} = [x(t_0 + \tau/T), x(t_0 + 2\tau/T), \dots x(t_0 + \tau)]$ and acquires a corresponding stream of observations $\mathbf{z}_{t_0} := [z_{t_0+\tau/T}, z_{t_0+2\tau/T}, \dots, z_{t_0+\tau}]$. For the observation stream $\mathbf{z}_{t_0}$ acquired at positions $\mathbf{x}_{t_0}$ we have the observation model $p(\mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}, \mathbf{m})$, the joint distribution $p(\mathbf{m}, \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}) = p(\mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}, \mathbf{m})p(\mathbf{m})$ [cf. (5.2.1)] and the marginal distribution $p(\mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}) = \sum_{\mathbf{m}} p(\mathbf{m}, \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0})$ [cf. (5.2.2)]. The CSQMI between the map $\mathbf{m}$ and the observations $\mathbf{z}_{t_0}$ acquired by following the trajectory $\mathbf{x}_{t_0}$ is defined as

$$\mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\right] = -\log \frac{\left\langle p\big(\mathbf{m}, \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\big), p\big(\mathbf{m}\big)p\big(\mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\big)\right\rangle^2}{\left\|p\big(\mathbf{m}, \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\big)\right\|^2 \left\|p\big(\mathbf{m}\big)p\big(\mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\big)\right\|^2}. \tag{5.2.4}$$

In the definition in (5.2.4), the product $p\big(\mathbf{m}\big)p\big(\mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\big)$ represents the joint probability distribution of the map $\mathbf{m}$ and the observations $\mathbf{z}_{t_0}$ when map and observations are independent. Thus, the ratio in (5.2.4) measures how far from independence the variables $\mathbf{m}$ and $\mathbf{z}_{t_0}$ – with joint probability distribution $p\big(\mathbf{m}, \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\big)$ – are and, in that sense, how much information we expect to gain about the map $\mathbf{m}$ from following the trajectory $\mathbf{x}_{t_0}$ and acquiring the observations $\mathbf{z}_{t_0}$. It is then reasonable to require the robot to follow the trajectory that maximizes $\mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\right]$, which we formally define as the solution of the optimization problem,

$$\begin{aligned} \eta^* = \max_{\mathbf{x}_{t_0}} \quad & \mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,|\, \mathbf{x}_{t_0}\right] \\ \text{s. t.} \quad & x(t) = x(t_0) + \int_{t_0}^{t} \dot{x}(v)\, dv \\ & \mathbf{x}_{t_0} = [x(t_0 + \tau/T), x(t_0 + 2\tau/T), \dots x(t_0 + \tau)]. \end{aligned} \tag{5.2.5}$$

The solution of (5.2.5) is a trajectory $\mathbf{x}_{t_0}$ that, if followed by the robot, achieves an information

gain η^*.

Information Theoretic Explorer

The purpose of the Information Theoretic Explorer is to plan an optimal trajectory for the robot as it explores the environment. For this we use a recent advancement in information theoretic mapping, namely Cauchy-Schwarz Quadratic Mutual Information (CSQMI) [14]. We begin with a robot located at $x_t = x(t)$ operating in environment $\mathbf{m}$. The robot is equipped with a 2 dimensional laser range finder mounted parallel to the floor, with B beams emanating radially from the sensor. Each beam has a minimum range, z_{min}, and maximum range, z_{max}. Consider then, a single beam b with measurement noise σ_s^2, and define d_t^b as the distance to the first obstacle along that beam. The probability distribution for the sensor measurement is then,

$$p(z_t^b = z \,|\, x_t, \mathbf{m}) = p(z_t^b = z \,|\, d_t^b) = \mathbf{N}(z; d_t^b, \sigma_s^2), \tag{5.2.6}$$

where $\mathbf{N}(x; \mu, \sigma^2)$ is the likelihood of drawing the value z from a Normal distribution with mean μ and variance σ^2. As the ray cast by the beam intersects with cells in $\mathbf{m}$, we define those cells as the set $\mathbf{c}$. Since (5.2.6) is entirely dependent on the first obstacle along the beam, we note that only the cells in $\mathbf{c}$ need to be considered. Therefore conditioning the measurement on $\mathbf{c}$ is equivalent to conditioning on $\mathbf{m}$ and x_t,

$$p(z_t^b \,|\, \mathbf{c}) = p(z_t^b \,|\, x_t, \mathbf{m}). \tag{5.2.7}$$

Next, we define the indicator function e_i, which takes the value of 1 when the i^{th} cell in $\mathbf{c}$ contains the first obstacle and 0 otherwise. We define the special case where $\mathbf{c}$ contains no obstacles as e_0. For a given beam, the probability that the i^{th} cell contains the first obstacle can be written as $p(e_i|x_t)$, which is obtained from $p(\mathbf{m})$. We can then compute the probability distribution of the

measurements given x_t, [cf. (5.2.2)]

$$p(z_t^b \,|\, x_t) = \sum_{\mathbf{m}} p\left(\mathbf{m}, z_t^b \,|\, x_t\right) = \sum_{i=0}^{C} p(z_t^b \,|\, e_i) p(e_i \,|\, x_t), \tag{5.2.8}$$

where $C = |\mathbf{c}|$. Using (5.2.7) we note the $\mathrm{I}\left[\mathbf{m}; z_t^b \,|\, x_t\right] = \mathrm{I}\left[\mathbf{c}; z_t^b \,|\, x_t\right]$. This now allows us to obtain the following approximation for CSQMI [14], from (5.2.4),

$$\begin{aligned}\mathrm{I}\left[\mathbf{m}; z_t^b \,|\, x_t\right] = \log \sum_{l=0}^{C} w_l \mathbf{N}(0; 0, 2\sigma_s^2) + \log \prod_{i=1}^{C} p_i \sum_{j=0}^{C} \sum_{l=j-\Delta}^{j+\Delta} p(e_j \,|\, x_t) p(e_l \,|\, x_t) \mathbf{N}(\mu_l; \mu_j, 2\sigma_s^2) \\ - 2 \log \sum_{j=0}^{C} \sum_{l=j-\Delta}^{j+\Delta} p(e_j \,|\, x_t) w_l \mathbf{N}(\mu_l; \mu_j, 2\sigma_s^2),\end{aligned} \tag{5.2.9}$$

where $p_k = \left(p\left(m_{k'}\right)^2 + \left(1 - p(m_{k'})\right)^2\right)$ and k' is the index in the map for the k^{th} cell of $\mathbf{c}$. The value of μ_j is determined by e_j and (5.2.6). The weights w_l for $0 < l < C$ are computed as follows,

$$w_l = p^2(e_l \,|\, x_t) \prod_{j=l+1}^{C} p_j, \tag{5.2.10}$$

with $w_0 = p^2(e_0)$ and $w_C = p^2(e_C)$. The inner sums in (5.2.9) are constrained to $l = j \pm \Delta$ due to the realization that 99.7% of the mass of a Gaussian distribution is within $\pm 3\sigma$. Therefore, $\mathbf{N}(\mu_l; \mu_j, 2\sigma_s^2) \approx 0$ when $|\mu_l - \mu_j| > 3\sqrt{2}\sigma$. Typically determining which μ_l are close to μ_j is difficult, but since the means are determined by the cells, we know that they are monotonically increasing with the cell index and thus easily identifiable. Additionally, we note that map cell lengths are slightly larger than the variance of the sensor. Therefore using $\Delta = 4$, implies that $\mathbf{N}(\mu_l; \mu_j, 2\sigma_s^2) \approx 0$ for all $|l - j| > \Delta$, thus the cells beyond $\pm\Delta$ can be removed from the double sum. This decreases the computational complexity of CSQMI to linear in the number of cells considered.

With (5.2.9) approximating the CSQMI of a single beam we extend the formulation to include a series of T robot poses where B sensor measurements are obtained at each pose. Between $t = t_0$

and $t = t_0 + \tau$, the robot moves through a series of poses $\mathbf{x}_{t_0} = [x(t_0 + \tau/T), \ldots x(t_0 + \tau)]$ and a collection of beams, $\mathbf{z}_{t_0} = [\mathbf{z}_{t_0+\tau/T}, \ldots, \mathbf{z}_{t_0+\tau}]$. Our goal is to determine trajectories that are maximally informative which requires the computation of $\mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,\middle|\, \mathbf{x}_{t_0}\right]$. The evaluation of this quantity is complicated be the possibility of two measurements not being independent. If all the measurements were independent as assumed in some systems [36, 41], the evaluation would be simply,

$$\mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,\middle|\, \mathbf{x}_{t_0}\right] = \sum_{z_t^b \in \mathbf{z}_{t_0}} \mathrm{I}\left[\mathbf{m}; z_t^b \,\middle|\, \mathbf{x}_{t_0}\right]. \tag{5.2.11}$$

Unfortunately, this assumption has been shown to result in gross over estimation of CSQMI. Therefore, we use a method for extracting a set of nearly independent measurements, $\mathcal{Z}$, from [14], and approximate the total CSQMI for a given trajectory as

$$\mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,\middle|\, \mathbf{x}_{t_0}\right] = \sum_{z_t^b \in \mathcal{Z}} \mathrm{I}\left[\mathbf{m}; z_t^b \,\middle|\, \mathbf{x}_{t_0}\right]. \tag{5.2.12}$$

Next, we define $D(\mathbf{x}_{t_0}) \leq \tau$ as the time needed to execute the trajectory. Using this and (5.2.12) the Information Theoretic Explorer solves this modified version of (5.2.5),

$$\begin{aligned} \max_{\mathbf{x}_{t_0}} \quad & \frac{\mathrm{I}\left[\mathbf{m}; \mathbf{z}_{t_0} \,\middle|\, \mathbf{x}_{t_0}\right]}{D(\mathbf{x}_{t_0})} \\ \text{s.t.} \quad & x(t) = x(t_0) + \int_{t_0}^{t} \dot{x}(v)\, dv \\ & \mathbf{x}_{t_0} = [x(t_0 + \tau/T), x(t_0 + 2\tau/T), \ldots x(t_0 + \tau)]. \end{aligned} \tag{5.2.13}$$

By maximizing the ratio of CSQMI with the duration of the trajectory we seek to balance the tradeoff between exploration and exploitation, i.e. the robot will choose shorter trajectories over longer ones if the expected information gain is comparable, [87].

5.2.2 Dynamic Communication Requirements

In addition to efficiently mapping the environment, the lead robot must also transmit the map data and the primary sensor's data back to the access point with minimal delay. In order to reduce the amount of data that needs to be transmitted, only the changes in the best estimate of the map are transmitted. In contrast to transmitting the full map on every update, the differential approach requires data transmission only when new information is obtained. This results in periods of low rates when the map is unchanged and periods of high rates when new, previously unexplored, areas are encountered. Since the primary objective of the team is to provide real-time situational awareness, the primary sensor's data flow is prioritized over the map updates. As such the update only utilize the residual bandwidth available after the primary sensor's data is transmitted. There are periods where the residual bandwidth is insufficient for the map updates, thus they must be queued and transmitted as bandwidth becomes available. Queueing the data introduces a delay between the update of the map at the leader and the update of the map at the access point. A delay in the map updates at the access point can be detrimental to the support robots since they are required for successful localization and channel estimation. Additionally, if the delay experienced by the map updates is large enough, any benefit of the lead robot quickly mapping the environment may be lost due to the data taking longer to reach the access point, as a result of minimal residual bandwidth.

The flow of the primary sensor and the map updates are combined into a single flow since they are both destined for the access point. While the system is capable of handling the two flow separately there is no difference in the results, thus the two flows are combined for the sake of clarity. During operation the experienced end-to-end rate for the primary sensor's flow is $a^e_{pri} = \min\{a^e, a_{pri}\}$, where a^e is the currently available end-to-end rate and a_{pri} is the primary sensor's required rate. The remaining bandwidth, $a^e_{map} = a^e - a^e_{pri}$ is allocated to the map updates.

For the system to satisfy the original communication requirement the minimum desired rate

must be sufficient for the primary sensor's flow, $a^1_{l,m} > a_s$. Using this requirement will allow the system to satisfy the primary sensor, but the delays to the map updates can be arbitrarily large. To avoid this, the value of $a^1_{l,m}$ can be increased so that the maximum expected delay of a map update is below some threshold, d_{map}. This can be achieved by estimating the maximum amount of change between sequential maps and determining the number of packets, n_{map} necessary to transmit that information. This leads to the requirement that the time needed to transmit n_{map} packets at a rate of r_{map} be below the threshold, $d_{map} \geq n_{map}/r_{map}$. Therefore, the value of r_{map} must be chosen such that this requirement is satisfied. Since the communication channels are capable of transmitting r_p packets per seconds, the required residual end-to-end rate can be computed from the end-to-end packet rates as $a_{map} = r_{map}/r_p$. The additional rate, a_{map}, combined with the primary sensor's rate, a_{pri}, becomes the new value of $a^1_{l,m} = \min\{1.0, a_{map} + a_{pri}\}$. It can be seen that as the maximum tolerable delay d_{map} is decreased, the value of $a^1_{l,m}$ increases which restricts the total area that can be explored by the leader. This is due to the maximum separating distance between two robots being inversely proportional to $a^1_{l,m}$. This approach also suffers from the drawback of applying conservative communication requirements even when there is minimal new information, such as revisiting a previously mapped location.

These limitations lead to the desire to dynamically adjust $a^1_{l,m}$ based on the current communication conditions. This approachmore effectively balances the competing desire of mapping the largest area with minimizing the delay of the map updates at the access point. First, for the value of $a^1_{l,m}$ to dynamically adjust, the local controller must be adapted from its current form to allow for changes in $a^1_{l,m}$ without compromising network integrity. Since a decrease in $a^1_{l,m}$ can only increase the margin afforded to network integrity, due it relaxing the communication requirements, we only focus on times then the value is increased. The goal then is to guarantee that the value of $a^1_{l,m}$ will never increase enough to cause the current formation to become infeasible. To provide this guarantee, we begin by defining a path between two robots as a series of links that connect robots i and j, without repeating a link. Written explicitly,

$p_{i,j} = \{(i_p, j_p)\}_{p=1}^{P}$, where $i_1 = i$, $j_P = j$, $i_p = j_{p-1}$, $(i_p, j_p) \neq (i_q, j_q)$ for all $q \neq p$. Using the communication rate for the value of the link, $v(i_p, j_p) = R(x_{i_p}(t), x_{j_p}(t))$, we define the score of a path $s(p_{i,j}) = \min_p v(i_p, j_p) = \min_p R(x_{i_p}(t), x_{j_p}(t))$. This score can be interpreted as the rate at which data can be transmitted from robot i to robot j if only the path $p_{i,j}$ is used. Since there are multiple paths the data can travel, we define the set of all possible paths from robot i to robot j as $\mathcal{P}_{i,j}$. Continuing with the interpretation of the score of a single path, we define the score of the set $\mathcal{P}_{i,j}$ as the maximum score of its paths, $m_{i,j} = \max_{p_{i,j} \in \mathcal{P}_{i,j}} s(p_{i,j})$, which we refer to as the max-min path score. Therefore, given the current formation, we can see that the maximum end-to-end rate between robots i and j is $m_{i,j}$. Thus, as long as $a^1_{l,m} \leq m_{l,N}$, the change in the communication requirements will not make the current formation infeasible.

With access to only local information it is difficult for the leader to estimate its max-min path score. To overcome this without introducing global coordination, the local controllers maintain their current max-min path score with the access point, $m_{i,N}$, in a distributed manner. To compute $m_{i,N}$ we begin by constructing a subgraph of the current communication graph. The nodes in this graph are robot i, the access point, and only the neighbors that robot i can send data to. To determine this set of neighbors we compare their dual variables with robot i, specifically only those in which $\lambda^k_i > \lambda^k_j$. It can be seen when $\lambda^k_i < \lambda^k_j$, the optimal solution to (4.1.13) will include $\alpha^k_{ij} = 0$. Thus, robot i will not transmit any received data to robot j and robot j can be ignored in the subgraph. In the subgraph, the value of the links between robot i and its usable neighbors is the current estimate of the channel rate $R(x_i(t), x_j(t))$, and each neighbor has a link back to the access point with a value equal to its current max-min path score, mj, N. Robot i then computes its max-min path score over the subgraph to compute its current $m_{i,N}$. These values are computed and exchanged much like the dual variables, but at a much lower rate. This process results in an accurate estimate of the max-min path score for each robot, which can be used by the lead robot to limit the value of $a^1_{l,m}$ and guarantee that the formation will never become infeasible. Since this value is computed in a distributed manner there will be a delay in the propagation of

information to the leader. Thus, we introduce a tolerance $\epsilon > 0$, that provides a margin of error when comparing the value of $m_{l,N}$ with the requested value of $a^1_{l,m}$. When the requested value for $a^1_{l,m}$ is larger than $m_l - \epsilon$, $m_l - \epsilon$ is used, otherwise the requested $a^1_{l,m}$ is used.

With the ability to dynamically adjust the value of $a^1_{l,m}$, we shift our focus to the development of a mechanism that will compute the desired $a^1_{l,m}$. For this computation we derive inspiration from back pressure routing, originally developed in [82]. In this system, the size of the transmit queue is used to exert pressure on the communication requirements, namely the larger the queue size the more conservative the communication requirements must become. Thus, we return to the acceptable level of delay for the map updates, d_{map}. The value of d_{map} can also be interpreted as the maximum time a packet should remain in the transmit queue. Thus, the value computed for a_{map} should allow for the queue to be cleared in d_{map} seconds. Using this interpretation, along with the current size of the queue, q_{map} and the nominal packet per second rate of the communication channel, r_p, we can compute the end-to-end rate required to clear the queue as $a_{map} = q_{map}/(r_p d_{map})$. This formula for a_{map} provides the desirable properties of $a_{map} = 0$ when the queue is empty, linear growth with the queue size, and maximum communication requirement when the queue size is very large.

Since the map updates are discrete in time, the number of packets in the queue can increase dramatically when a new area is encountered. To mitigate these large jumps, the value requested for $a^1_{l,m} = \min\{1.0, a_{pri} + \hat{a}_{map}\}$, where $\hat{a}_{map}$ is the output of a low pass filter with input a_{map}. The result of these modifications is the teams ability to dynamically adjust to the size of the transmit queue, while avoiding the imposition of requirements that are infeasible for the given formation.

5.2.3 Integration

To complete the integration, we note that the series of poses that constitute a trajectory, $\mathbf{x}_{t_0}$, in the Information Theoretic Explorer are interchangeable with the trajectories provided to the local

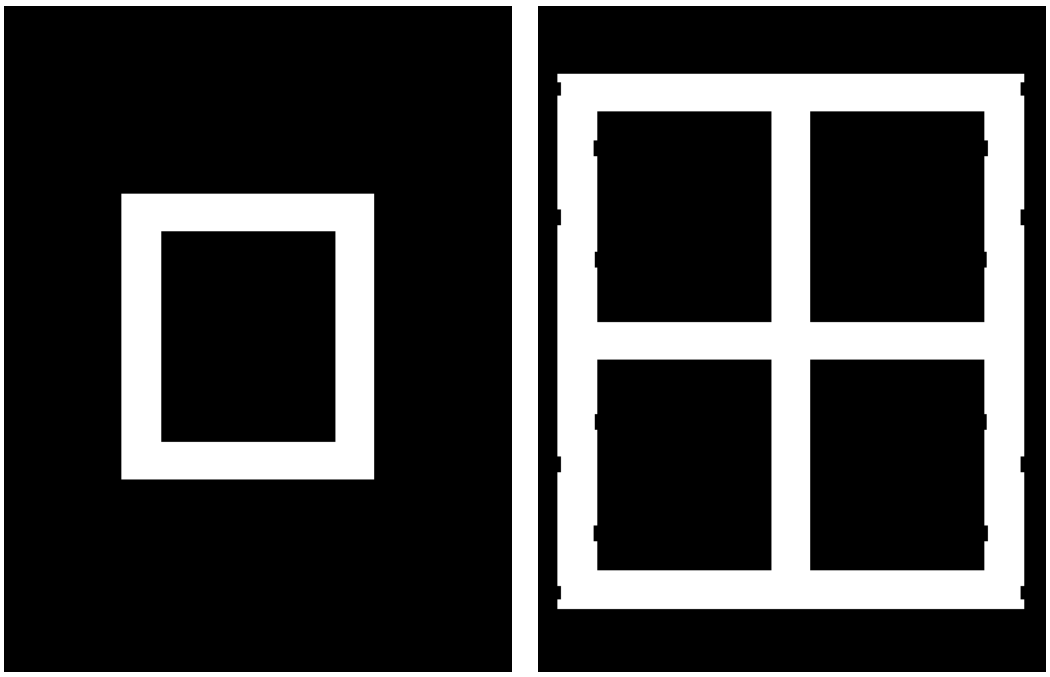
controllers, $\mathcal{X}_l$. Thus, the resulting trajectories can be ingested by the local controllers with no adjustments.

With the Information Theoretic Explorer able to determine trajectories that provide the maximum rate of information gain, we have now detailed all of the major components of the SCLAM system. One of the byproducts of (5.2.13) is the estimated information gain η^* that results from following the trajectory, which we can use to indicate when the environment has been sufficiently mapped. Thus, we define η_m as the minimum amount of information gain that the system is willing to act upon, and halt the mapping process when $\eta^* < \eta_m$. The level of η_m chosen determines the uncertainty in the map that we are willing to accept. Setting η_m low will result in a very confident map, but may require more time. Conversely, setting η_m high will result in decreased execution time, but the resulting map may have high areas of uncertainty. With this, the description of the autonomous SCLAM system is complete, and we shift our focus to validation of the system.

5.3 Experimental Configuration

5.3.1 Environments

In this chapter there are four environments used for simulation, of which two are used for experimentation. The two environments used for simulations and experiments are the same as those used in previous chapters, namely the 6^{th} floor of Levine and the 5^{th} floor of Levine-GRW. The change in floor from the 5^{th} to the 6^{th} does not affect consistency since they are nearly identical in layout and construction; thus, they are interchangeable. The two environments that are used solely for simulation were constructed for differing reason; one for simplicity shown in Fig. 5.3a, called Small-Loop, and the other for complexity shown in Fig. 5.3b, called Grid. The Small-Loop environment was constructed from a single loop from figure eight layout of Levine-GRW, and the Grid environment is Levine-GRW mirrored along the vertical axis. These two environments combined with the representations of Levine and Levine-GRW make up the collection of simulation



(a) Small-Loop (b) Grid

Figure 5.3: New simulation environments.

environments. For the purpose of this chapter it is useful to rank these environments in terms of increasing complexity. Starting with the simplest, Small-Loop, to the Levine, and Levine-GRW, to the most complex, Grid, with each of these covering 60, 135, 143, and 178 m^2 respectively.

5.3.2 System Parameters

In this chapter, the model used for the the wireless channel estimates between two robots, $\hat{R}(x_i(t), x_j(t))$ is a polynomial fitting of experimental curves found in the literature [2], using the most recently published $\hat{\mathbf{m}}(t_p)$. For the computation of CSQMI, we model the Hokuyo laser scanner as a plane laser, with 200 distinct beams covering the 270° field of view. We use $z_{min} = 0.1$ m and $z_{max} = 8.0$ m for the minimum and maximum sensor ranges, with noise, $\sigma_s = 0.03$. The system signals that the environment has been sufficiently mapped when the information gain of the remaining poses falls below $\eta_m = 4$, a very low value for this configuration, as seen in the results. To create the thresholded map we use $\nu_{free} = 0.20$ and $\nu_{obs} = 0.65$. For all of the simu-

lations and experiments, the team consisted of three mobile robots and one access point, $N = 4$. The empirical end-to-end data rate is measured by the percentage of packets transmitted over the ad-hoc network via UDP and received at the access point from the lead robot.

5.4 Fixed Communication Requirements

This section is focused on validating the ability of the SCLAM system to concurrently map the environment and provide the minimum desired end-to-end rate. For this section the transmission of the map updates is not included in the communication requirements, thus $a^1_{l,m}$ is fixed. In Section 5.5 the map updates are included in communication requirements and a dynamic $a^1_{l,m}$ is used.

5.4.1 Simulations

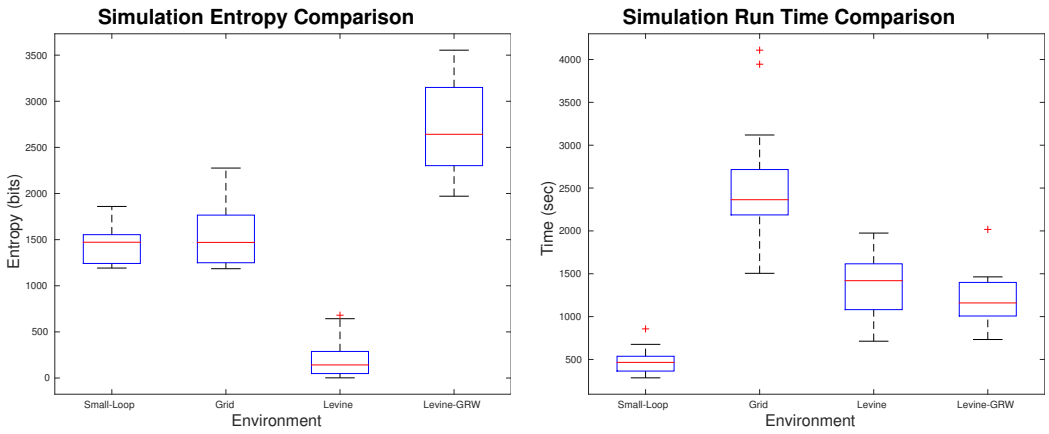
To examine the behavior of the system in various environments, we ran a series of simulations in the environments described in Section 5.3.1. The goal in these simulations was to map the environment to the required level of accuracy in as little time as possible, while still maintaining the desired level of end-to-end data rate between the lead robot and the access point. The starting formation is held constant for each trial in a given environment and the minimum end-to-end rate back to the access point is set at $a^1_{l,m} = 0.3$. When the Information Theoretic Explorer indicates that there are no more trajectories with adequate information gain, the simulation is halted and the final map is examined. Since the main responsibility of the system is to map the environment, we compute the uncertainty in the cells of the final generated map that correspond to the observable cells in the environment. Removing all the unobservable cells for an environment provides a normalized measure of the resulting maps which can be used to compared results across environments. To compute the uncertainty of a cell we use Shannon entropy [15], $H[m_k] = -p(m_k)\log_2(p(m_k)) - (1 - p(m_k))\log_2(1 - p(m_k))$. A cell will have an entropy of 0 if the probability of that cell containing an

obstacle is either 0 or 1, i.e. there is no ambiguity, and it will have a maximum entropy of 1 if the probability is 0.5, i.e. unobserved. Summing over all observable cells $H[m] = \sum_k H[m_k]$ provides the entropy of the map. The lower this value the less uncertainty associated with that map and thus a more accurate representation of the environment. For reference the maximum entropy for the four simulation environments are 2.7×10^4 for the Small-Loop, 7.5×10^4 for the Grid, 5.7×10^4 for Levine, and 6.0×10^4 for Levine-GRW. The resulting entropies of the simulations are plotted in Fig. 5.4a. As we can see from the figure, the average remaining entropy in the maps is a fraction of the maximum, which indicates that the system sufficiently mapped the entire area.

Next, we plot the time required to complete the map for the given environment in Fig. 5.4b. There are a few items to note in from this plot. First, as expected the time needed to map the Small-Loop environment is small and tightly clustered. Second, the amount of time needed to map the two representative environments is comparable. This is an expected result since the area explored is approximately equal. The interesting result is the variation in the time needed for Levine. This is due to the planner taking longer to find feasible trajectories, which is due to the needs to backtrack around the loop to reach some destinations. This is also present in Levine-GRW, but since the loops are smaller, the amount of time needed to find a path is less. Finally, we notice the large variability in the grid simulation environment. As with Levine-GRW, this environment requires the global planner to generate trajectories that backtrack substantially. These simulations confirm that the system is able to successfully map complex environments while preserving the required data rate.

5.4.2 Experimental Evaluation

In this section we focus on demonstrating the ability of our system to operate in the real world. These experiments show that our system is capable of mapping and operating in a previously unknown environment. For these experiments the *Scarab* platform [48] is used for both the lead and support robots, and the XBee radios are used for the ad-hoc multi-hop wireless network. The



(a) Distribution of entropies of the final map for each set of trials in the four environments.

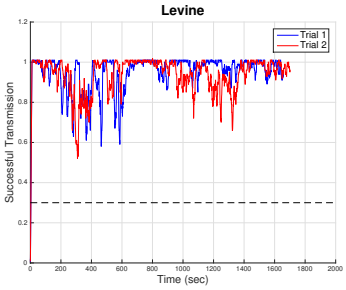
(b) Distribution of runtimes for multiple simulations in various environments.

Figure 5.4: Metrics for the multiple trials in each environment.

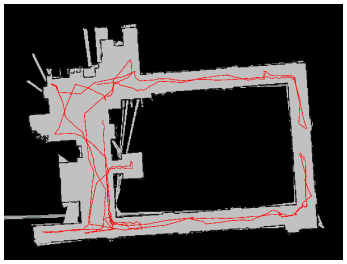
two environments used in this section are the 6^{th} floor Levine and the 5^{th} floor Levine-GRW. The parameters used for the Information Theoretic Explorer and communication requirements are identical to those used in the previous simulations.

In the Levine environment, the team's initial formation is located in the upper left corner of the environment. Two separate trials were performed and the resulting end-to-end data rates experienced by the lead robot are plotted in Fig. 5.5a. The two trials are indicated by the blue and red lines, while the minimum desired rate is shown as a black dashed line. Notice that for both trials the data rates stay sufficiently above the minimum value. Also, note that contrary to the results in the simulations the two trials complete in approximately the same amount of time and fall on the border of the 75^{th} percentile of the simulation runtimes. In Figs. 5.5b and 5.5c we plot the resulting map and trajectory of the lead robot. Aside from the rotation, which is an artifact of the initial orientation of the robot not being identical between trials, the two maps a very similar. Notice that the trajectories followed by the lead robot are also similar. This is due to a similar evolution of the map as the experiments unfolded, and purely coincidental.

The next environment explored was the 5^{th} floor of Levine-GRW. In this environment the team's initial formation is in the upper left corner. Similar to the Levine experiment, two separate



(a) Data rates for two trials, shown in red and blue, in the Levine environment. The dashed black line represents the minimum required data rate.



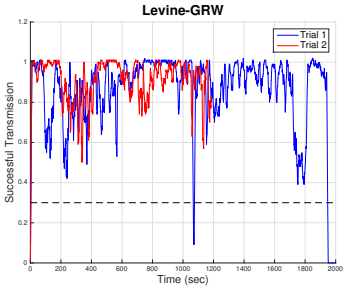
(b) Trial 1 map with trajectory followed by the lead robot in red.

(c) Trial 2 map with trajectory followed by the lead robot in red.

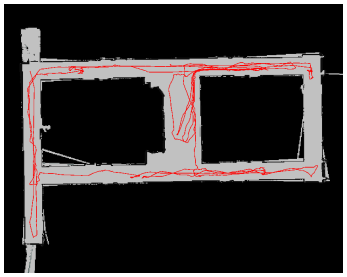
Figure 5.5: Results from experiments in Levine.

trials were performed and the resulting data rates are plotted in Fig. 5.6a. The data rate for the two trials are shown as blue and red solid lines, with the minimum data rate is shown as a black dashed line. The first item that is obvious is the drop off seen in the data rate for trial 1 around 1050 seconds. This drop off can be attributed to the distributed algorithm taking time to converge to the new optimal routing solution after line-of-sight between two robots is lost. Typically the system is able to absorb such shocks, but there are times when the convergence rate is insufficient and small drops in data rates are observed. Fortunately, these drops in performance are only temporary and are quickly recovered from. The other item to note is the large difference in the completion time between the two trials. The first trial takes almost 2000 seconds, while the second trial completes in about 1200 seconds. This difference is directly attributable to the CSQMI system alternating between two locations. This is seen in Fig. 5.6b where the middle open area is traversed repeatedly while in Fig. 5.6c does not exhibit this behavior. In comparison to the simulation runtimes the first trial is an outlier and the second trial is very close to the mean.

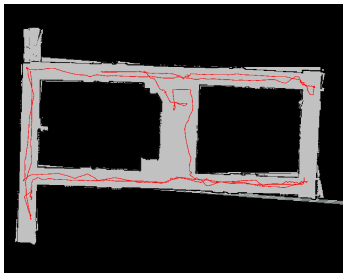
Next we compute the map entropy as a function of time for the 4 trials. Using the same process as in Section 5.4.1, but to determine which cells are observable, we use the final map of each trial as ground truth, and sum over all cells within 0.1 m of an observable cell. The results from all 4 trials are plotted in Fig. 5.7, with the blue and red lines corresponding to the Levine-GRW and



(a) Data rates for two trials, shown in red and blue, in the Levine-GRW environment. The dashed black line represents the minimum required data rate.



(b) Trial 1 map with trajectory followed by the lead robot in red.



(c) Trial 2 map with trajectory followed by the lead robot in red.

Figure 5.6: Results from experiments in Levine-GRW.

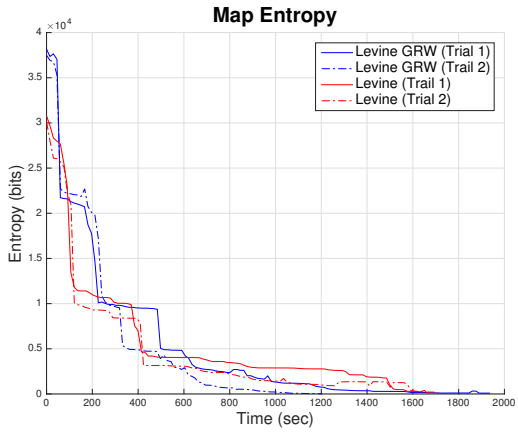


Figure 5.7: Map entropy for the 4 experimental trials.

Levine environments, respectively. For these two environments there are approximately 6.0×10^4 observable cells. As we can see in Fig. 5.7, there is sharp decrease in the map entropy in the first few hundred seconds. For all these trials, the large jumps in the entropy can be directly attributed to the lead robot entering an intersection, where a previously unobserved hallway becomes visible. Looking at the figure, we can also see that for three of the four trials the map's entropy does not decrease much beyond its value at 1000 seconds, due to only small refinements in the map. This is especially true for trial 1 in the Levine-GRW environment, the outlier with respect to time. The repeated traveling in the upper right area does not noticeably change the entropy of the map which indicates that η_m in the CSQMI system is very conservative.

5.5 Dynamic Communication Requirements

This section includes a series of simulations and experiments to demonstrate the benefit of dynamic communication requirements. In contrast to Section 5.4, the map updates are now included in the communication requirements. Since the ability of the system to map and provide the required end-to-end rate has been demonstrated, this section focuses on the timeliness of the map updates and the benefits of dynamic communication requirements.

5.5.1 Simulations

As in the previous section we begin with a series of simulations in the four environments, with a fixed and a dynamic $a^1_{l,m}$. For consistency with the previous results, the composition of the team, the starting formation and the information theoretic parameters are identical; only $a^1_{l,m}$ is modified. Since the map updates are now included in the communication requirements, the value of $a^1_{l,m}$ for the fixed scenarios was set to 0.5, to allow for a primary sensor rate of 0.4 and a map update rate of 0.1. For the dynamic scenario the value of $a^1_{l,m}$ was allowed to vary between 0.4 and 1.0, depending on the current conditions. The end-to-end rate was simulated by computing the max flow over a weighted graph, where the edge weights were the product of the routing solution and expected channel rate between the two robots. Since the end-to-end rate is shared between the map updates and the primary sensor, the remaining flow after subtracting the primary sensor's flow was used to remove items from the transmit queue. In each environment 15 simulations were run for both the fixed and dynamic settings, with the results plotted in Fig. 5.8. In these plots the red line corresponds to the fixed scenario and the blue lines corresponds to the dynamic scenario. The results of each environment are grouped by row from top to bottom: Small-Loop, Grid, Levine, Levine-GRW. For each environment there are three statistics plotted grouped by column from left to right: average transmit queue size at the leader, average cumulative number of packets created and received, and cumulative distribution function of the packet delay.

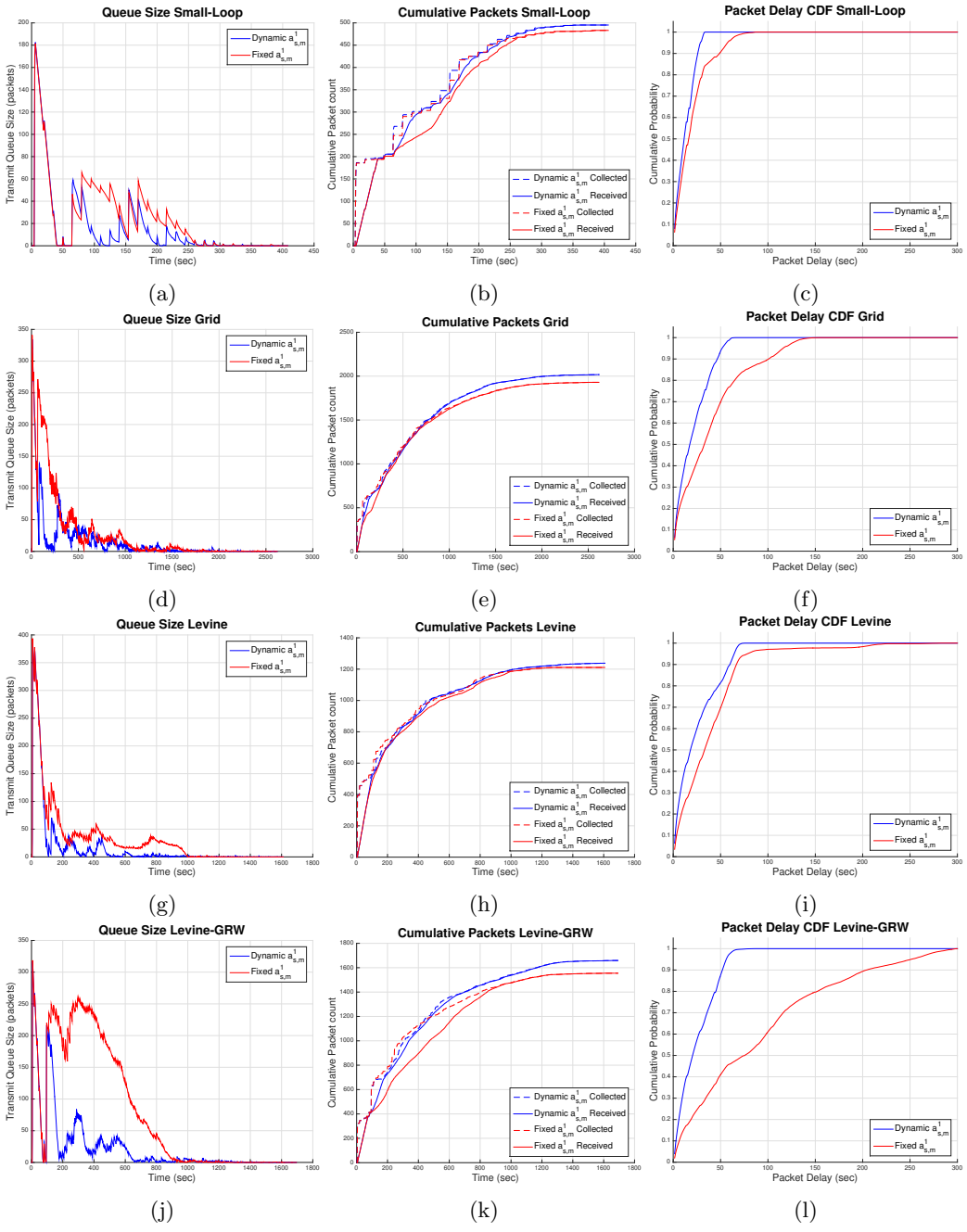


Figure 5.8: Simulation results comparing fixed communication requirements with dynamic requirements for the four environments considered.

We start by examining the first column of plots (Figs. 5.8a, 5.8d, 5.8g, and5.8j), which show the average number of packets in the transmit queue as a function of time. We note that when the queue size is small, the newly collected data is transmitted with less of a delay than if the queue size is large. With that in mind, we see that across all of the environments, the blue line is almost always below the red line, which indicates that throughout the deployment the average amount of data in the queue awaiting transmission is less for the dynamic scenario. Therefore, we know that newly collected data will experience less of a delay due to queueing in the dynamic scenario. Additionally, we can see that in the Grid and Levine-GRW environments (Figs. 5.8d and 5.8j) there are periods of time where the red line is much higher than the blue line, specifically between 150 - 300 seconds and 200 - 900 seconds, respectively. These large separations are due to large, previously unexplored, areas being encountered during exploration and the residual bandwidth being insufficient to clear the queue in the fixed scenario. Notice that this does not happen in the dynamic scenario because as the queue size grows, the communication requirements increases restricting the motion thus allowing the queue to clear.

The concern with temporarily restricting the motion of the team to allow for the queue to clear is that the total time required to complete the task will increase dramatically. Comparing the average completion time of the two scenarios for each environment shows that the average increase in the completion time was approximately 5%. This indicates that the dynamic adjustments only have a minor negative impact on task completion time. Next, we examined how the adjustments affect the rate of exploration. To evaluate this the cumulative rate of packet creation and reception as a function of time is plotted in the second column (Figs. 5.8b, 5.8e, 5.8h). In these plots there are two lines per color, a dashed and a solid line. The dashed lines track the creation of data at the leader, or the amount of map information collected up to a given time. Thus, the more quickly the line rises the faster the leader is mapping the environment. The solid lines track the reception of the updates at the access point, or the amount of map information that can be leveraged by the team at a given time. If the dynamic adjustments to the communication requirements were

detrimental to the rate of exploration, the red dashed line would increase at a much greater rate than the blue dashed line. As it is seen in the plots, for all four environments the rate of increase for the two dashed lines are in close agreement, indicating that the adjustments do not have a noticeable impact on the exploration rate. Surprisingly, the solid red line typically lags behind the solid blue line, indicating that in the dynamic scenarios the the access point is receiving map updates faster than in the fixed scenarios.

Finally, we examined the delay experienced by each packet for the two scenarios (Figs. 5.8c, 5.8f, 5.8i, and5.8l). Since these plots show the cumulative distribution function of the packet delay, the farther the curve is to the left, the smaller the expected delay experienced by any packet will be when it is received at the access point. As expected, we see that in all of the plot the blue line is to the left of the red line, indicating that the percentage of packets experiencing large delays is much less when the communication requirements are dynamic. This leads to the conclusion that the benefits of allowing dynamic communication requirements, namely dramatically less packet delay, outweighs the minor increase in task completion time.

5.5.2 Experimental Evaluation

As with Section 5.4.2, demonstrating the system operating effectively in simulation is insufficient, thus we ran another set of experiments to demonstrate the benefits of dynamic communication requirements during real operation. Again, three *Scarab* platforms and an access point operate in Levine and Levine-GRW. In these trials the parameters and initial formations from the simulations were used, and each of the two scenarios, fixed and dynamic, were executed. The resulting four data collections, two for Levine and two for Levine-GRW, are plotted in Fig. 5.9. As with the simulation results, the two environments are grouped by row and the different statistics are grouped by column. Examining the first column of plots (Figs. 5.9a and 5.9d), we see that the general behavior is consistent with the simulations. The average queue size for Levine is similar for both the fixed and dynamic scenarios, but the differences become apparent in Levine-GRW.

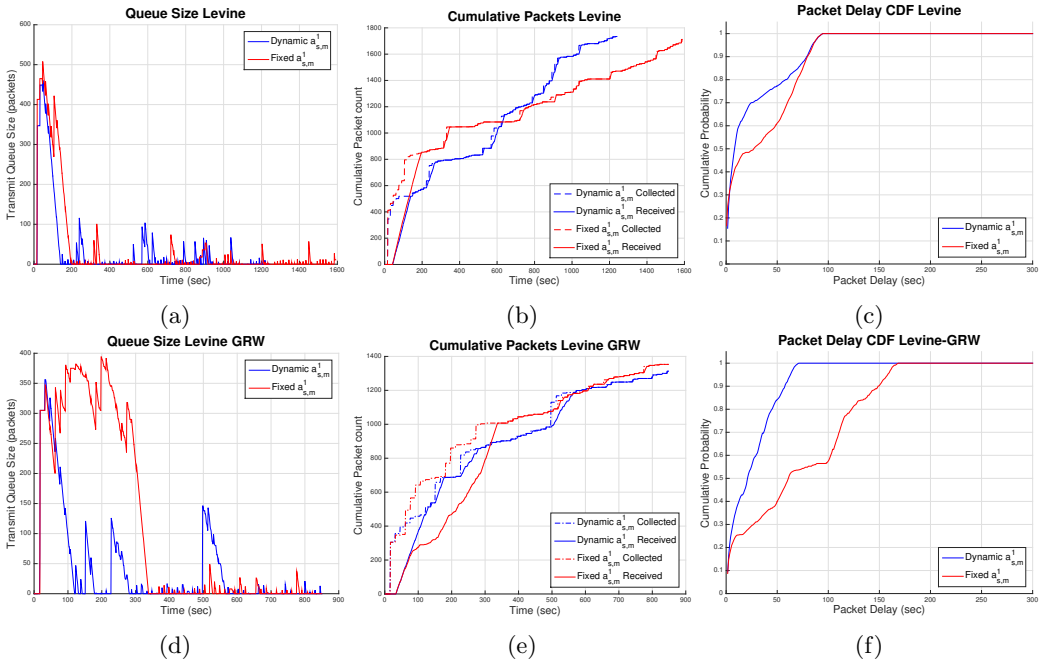
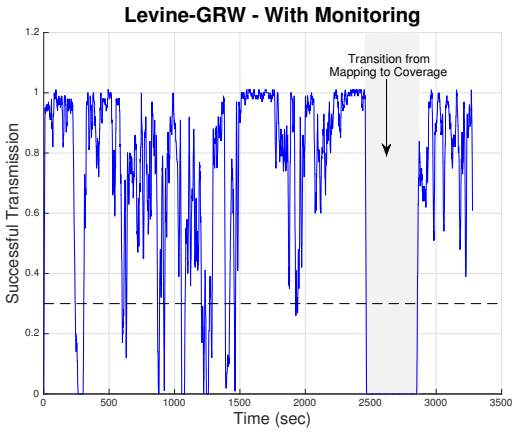


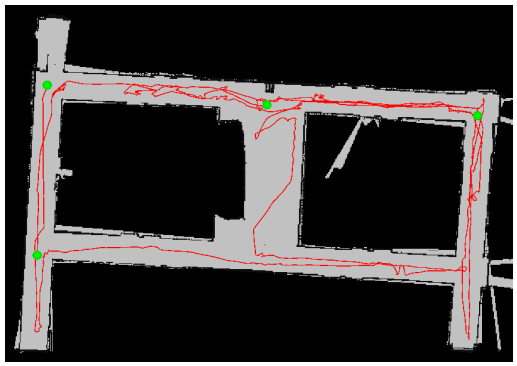
Figure 5.9: Experiment results comparing fixed communication requirements with dynamic requirements in Levine and Levine-GRW

As noted in the simulation section, the reason for the large queue size in the fixed scenario is the communication requirements, and thus motion, not reacting to the large influx of map information obtained when the central corridor is encountered. This results in a large backlog of packets in the leader's transmit queue, which takes longer to clear due to the smaller residual bandwidth.

In the next column of plots (Figs. 5.9b and 5.9f), the cumulative packet counts are plotted. In contrast to the simulation results, the dotted lines are not in agreement. This is due to the results in Figs. 5.8a, 5.8d, 5.8g, and5.8j plotting the average values and not one specific trial. Even with the disagreement between the two lines, it can still be seen that the rates are not dramatically different. In fact, for Levine the dynamic scenario completed in less time than the fixed one. Finally, the CDF of the packet delay is plotted in the final column, (Figs. 5.9c and 5.9f). As expected, the blue line is to the left of the red line indicating that overall the dynamic requirements reduces the experienced packet delay.



(a) Data rate in the Levine-GRW environment. The values before the shaded region are from the lead robot to the access point. The values after the shaded region are from the quad-rotor to the access point, while flying in a figure eight pattern twice. The dashed black line represents the minimum required data rate.



(b) Final map of Levine-GRW. The trajectory of the lead robot is indicated by the red line and the final location of the 3 robots and access point are indicated by the green circles.

Figure 5.10: Levine-GRW experiment with a highly mobile aerial platform patrolling after the autonomous SCLAM system successfully mapped the environment.

5.6 Map and Patrol an Unknown Environment

In the final experiment, we demonstrate the flexibility of our system to autonomously map an unknown environment and then transition into a support network so a highly mobile aerial platform can patrol the newly mapped environment. In this demonstration the Levine-GRW environment was chosen. As with the previous experiments, the team begins in the upper left corner of the environment and proceeds to map the environment while maintaining the end-to-end link between the lead robot and the access point. After the Information Theoretic Explorer determines the map is sufficiently complete, the team transitions to a support network with the primary objective of relaying communication from a patrolling quad-rotor back to the access point. In this network, all the robots are considered identical, since from a communication perspective their capabilities are all equivalent. The desired data rate back to the access point for the quad-rotor is 0.3. After the system determines the optimal formation, based on the communication model and assumed height of the quad-rotor, the robots assume the optimal formation. In Fig. 5.10a we plot the

incoming data rate at the access point. During the mapping phase the data is coming from the lead robot and during the coverage phase the data is originating from the quad-rotor. The delineation between the two phases is indicated by the shaded region during which time there is no data being generated. Notice that for the majority of the mapping phase the data rates are in excess of the minimum desired value, and during the coverage phase the rate is always above the minimum. In Fig. 5.10b we show the final map along with the trajectory, shown in red, followed by the lead robot. Also included in this figure is the final formation used for the coverage phase, indicated by the 4 green circles.

5.7 Summary

In this chapter, we present a system that autonomously maps an unknown environment while maintaining a required end-to-end data rate between robots on a team. We achieve this by first detailing the necessary modification to the components of the hybrid system that rely on a map of the environment. Then, we invert the hybrid architecture to allow for the local controller to assume a more pronounced role in the operation of the system. This allows the system to operate in an almost entirely distributed manner, only relying on global coordination when the team is trapped in a local minima. To complete the new requirement we integrate an Information Theoretic Explorer to select trajectories that are maximally informative of the environment. The Information Theoretic Explorer is able to provide adequate direction to the lead robot while the local controller, continuing to maintain the ad-hoc network, drives the robots through the environment. Additionally, we note that if the situational awareness task is to only map the environment, then the Information Theoretic Explorer can be run on the lead robot resulting in an almost completely distributed SCLAM system. The system is capable of supporting both the primary sensor's data flow as well as the map updates via dynamic communication requirements, which result in minimal delay of the received data.

This system is verified in both simulation and experiments. In the simulations we highlight the consistency and accuracy of the resulting maps. In the experiments we verify the ability to construct these maps while still maintaining the end-to-end link between the lead robot and the access point with minimal delay. We conclude with a demonstration of the team autonomously mapping an environment and then transitioning to complete another situational awareness task. Specifically, supporting a highly mobile aerial robot as it patrols the recently mapped environment.

Chapter 6

Conclusions

In this thesis, we addressed the problems related to a team of autonomous robots providing real-time situational awareness in unknown complex environments. We demonstrate our solutions to these problems though numerous simulations and experiments, with a focus on realistic scenarios. The main accomplishment of the work in this thesis is a hybrid system approach that solves the real-time situational awareness problem in an unknown environment with minimal global coordination. The system accomplishes this by systematically decomposing the problem into subproblems that can be solved in a distributed manner. To efficiently solve the subproblems, the system utilizes an adaptive communication model of the point-to-point communication links between robots while concurrently solving the motion planning and network routing problems. This hybrid system approach to the problem allows for the system to adapt to the specific task with minimal configuration changes and is thus applicable to a wide range of situations that were previously out of reach.

6.1 Summary of the Thesis

In Chapter 2 we detail a real-time situational awareness problem, in which a team of mobile robots is required to move through a complex environment while maintaining the desired end-to-end data rates between designated pairs of robots. To accomplish this, we formalize a concurrent mobility and routing problem by combining motion control and network integrity into a single optimization problem. We then examine the various ways to formulate the network integrity constraints and their effect on the resulting solutions. Next, we provide a taxonomy of the design space when solving this problem, allowing us to classify the systems based on their level of required coordination and the optimality of the resulting solution. Using this, we examine the existing systems proposed to solve the concurrent mobility and communication problem and extract the properties and features that allow us to meet our requirements. We conclude with a description of the robotic platforms, software framework, and primary environments used in this thesis.

In Chapter 3, we develop a lightweight confirmation protocol specially designed for use with robust routing solutions the over type of ad-hoc networks seen in mobile robot teams. We begin by demonstrating the benefits derived from a robust formulation of network integrity constraints over a non-robust formulation, through experiments in multiple locations and various team sizes. Next, we study the expected behavior of TCP and UDP, two ubiquitous communication protocols, over ad-hoc wireless networks, and determine that they are insufficient. This leads to the construction of MCTP, which is a protocol design specifically for use over ad-hoc wireless networks created by low-cost low-power transceivers found on mobile robotic platforms. The benefits of this protocol are demonstrated through simulations and experiments in which the successful transmission of both data and confirmation messages are compared to traditional protocols. We conclude with an experimental demonstration of MCTP providing near loss-less transmission of data from a sensing robot to an access point in a realistic environment over an ad-hoc network created by other mobile robots.

In Chapter 4, we develop a hybrid system capable of controlling a team of arbitrary size through a complex environment while completing the real-time situational awareness task. This system consists of a two stage feedback architecture where the outer loop is responsible for the initial global coordination and the inner loop is responsible for the motion and network routing of the specific robots. The plan developed by the outer loop is used as a roadmap by the inner loop to allow for a mostly distributed operation while avoiding local minima

Through simulation we demonstrate that the hybrid approach is able to operate successfully in complex environments in which a distributed system is insufficient. Additionally, the ability of the system to scale with the number of robots is shown in a scenario where a team of 25 robots is tasked with operating in a large complex environment. The system is then empirically compared to a recent centralized system to demonstrate that the dynamic distributed nature of the hybrid system provides greater performance than the centralized system without the coordination overhead. The dynamic nature of the hybrid system is highlighted further in an experiment where one of the support robots has a temporary motor failure. This failure causes the centralized system to lose network integrity while the hybrid system adapts accordingly

Next, we demonstrate the hybrid system's ability to complete the complex real-time situational awareness task of patrolling a series of hallways. First, we show that even when using a non-confirmation based communication protocol, UDP, the system is able to maintain the desired end-to-end data rates for the duration of the exercise, Finally, with the addition of the confirmation protocol developed in Chapter 3, MCTP, the system provides a near loss-less communication link for a sensing robot back to the access point via the ad-hoc network, throughout the deployment.

In Chapter 5 we augment the system developed in Chapter 4 so that it can effectively operate in unknown environments. To accomplish this, we begin by examining how incomplete information affects the various components of the system so that we can then modify the necessary components to allow for operation in such situations. Then, with the realization that the initial hybrid architecture would require increased global coordination when operating with dynamic goals, we

invert the architecture to rely more heavily on the local controllers and only utilize the global planner when necessary. With the new architecture we integrate a recently developed Information Theoretic Exploration system that determines the optimal trajectories of a mapping robot so that the environment can be mapped as efficiently as possible. We then establish this ability in environments of varying complexity through simulation and experiments. Finally, we demonstrate that a team of autonomous robots placed initially in an unknown environment, is able to construct an accurate map while maintaining network integrity and then assume strategic locations so that a highly mobile aerial platform can patrol the newly mapped environment.

6.2 Main Contributions

In this thesis, we address the problems faced by a team of robots tasked with providing real-time situational awareness and we make several key contributions. One contribution is the development, design, and implementation of a lightweight reliable communication protocol, MCTP, that is able to operate effectively over ac-hoc wireless networks. This ability is enabled by exploiting the structure of robust routing solutions as well as the link diversity present in mobile robot communication networks. Specifically, the ability for a packet to travel a different path if a drop is identified allows the protocol to mitigate the random nature of wireless links as well as the dynamic network topology of the communication network. In addition to the reliability provided by MCTP, this protocol is designed for use in teams with many robots. This results in a more efficient use of the confirmation channel than traditional protocols. The result is a communication protocol that is able to provide reliable communication over an ad-hoc wireless network constructed by low-cost, low-power transceivers.

Another key contribution is a decomposition of the real-time situational awareness problem. This provides the ability to discuss the proposed solutions in a 2-dimensional space, in which the two design parameters are optimality of the solution and required level of coordination, as opposed

to the typical centralized versus distributed approach. Using this high space, we are able to extract the properties that are beneficial to the situational awareness problem, namely global optimality and distributed coordination. This leads to a hybrid systems approach that produced a system utilizing highly capable distributed controllers to complete an arbitrary task with minimal global coordination. Furthermore, we demonstrate empirically the system's ability to obtain an optimal end-to-end data rate and to provide reactive control to unexpected events. We also demonstrate the ability of the system to scale through simulation.

The final contribution of this thesis is the extension of the real-time situational awareness task to successful operation in unknown environments. By enhancing the individual components and adding mild restrictions on their operation, we are able to allow for the team to safely operate as the environment is being mapped. To accommodate the dynamic nature of the task, the system architecture is augmented to increase the reliance on the local controllers, with additional safeguards. With this increased reliance, the system is now able to operate almost entirely in a distributed manner, while still providing globally optimal results. Not only did we modify the system, but we also integrated a state of the art Information Theoretic Exploration system that leverages Cauchy-Schwarz Quadratic Mutual Information to select optimal trajectories that map the environment in a highly efficient manner. This approach's versatility is demonstrated by the team's ability to begin in an unknown environment, map it and then complete other traditional situational awareness tasks.

While the scenarios considered in this thesis revolve around a single robot providing situational awareness, the methods and systems proposed are not limited to a single data flow. For example, the construction of the routing subsystem is specifically designed to operate when network integrity is composed of an arbitrary number of data flows. This allows for operation when multiple sensor robots are on the team.

6.3 Future Work

Even though we have experimentally demonstrated a system that is able to complete a real-time situational awareness task in unknown environments, several areas that might still be addressed to provide an even more robust system. In this section we identify possible avenues to explore in regards to increasing the viability of this system.

In Chapter 4 we developed a local controller that is able to successfully drive a robot to a goal location while preserving network integrity. While this controller is able to avoid local minima with the help of the global planner, its ability to independently avoid local minima is absent. Given that the final system relies heavily on the local controllers, any increase in its ability to operate in complex environments would greatly enhance the overall system. This is an area where large benefits could be realized, but it is also one of the most difficult to extend if only local information is available. One possible modification is to include specific information from other robots in the dual variable exchange, thus propagating that information through the multi-hop network. The information must not be added arbitrarily since the transmission of that data adds to the coordination overhead. Therefore, the information must be carefully chosen.

Another area of possible extension is to the system developed in Chapter 5. In this system only one robot is used for mapping, while the others are required to provide support. This limitation can be overcome by collecting all the maps from the robots in a single location and fusing them together to provide a larger map, which is an active research area. Since the Information Theoretic Explorer is capable of generating trajectories for multiple robots that maximize information gain, and the Hybrid System is capable of supporting multiple data flows, this extension is feasible with minimal system modifications.

Finally, the most obvious extension is to remove the requirement that the robots be ground platforms. While an aerial platform was utilized in the experiments, it was not being controlled by our distributed controller. To achieve integration of non-ground robots, the local controller

would have to be augmented to control robots with much more complicated dynamics than those considered in this thesis. Additionally, the communication modeling would have to extend into the third dimension. This is the course of action that will most likely result in the largest gains, since by leaving the ground plane the team will be able to explore and sense in locations that are unreachable by ground platforms.

Bibliography

[1] Aggarwal, A., Savage, S., and Anderson, T. Understanding the performance of tcp pacing. In *Proceedings of IEEE International Conference on Computer Communications* (2000), vol. 3, IEEE, pp. 1157–1165.

[2] Aguayo, D., Bicket, J., Biswas, S., Judd, G., and Morris, R. Link-level measurements from an 802.11b mesh network. *SIGCOMM Computer Communication Review 34*, 4 (Aug. 2004), 121–132.

[3] Al-Jubari, A. M., Othman, M., Ali, B. M., and Hamid, N. A. W. A. Tcp performance in multi-hop wireless ad hoc networks: challenges and solution. *EURASIP Journal on Wireless Communications and Networking 2011*, 1 (2011), 1–25.

[4] Antonelli, G., Arrichiello, F., Caccavale, F., and Marino, A. Decentralized time-varying formation control for multi-robot systems. *The International Journal of Robotics Research 33*, 7 (2014), 1029–1043.

[5] Ascending technologies inc. `http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-hummingbird/`, 2015.

[6] Betke, M., and Gurvits, L. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation 13*, 2 (Apr 1997), 251–263.

[7] Bourgaul, F., Makarenko, A., Williams, S. B., Grocholsky, B., Durrant-Whyte, H. F., et al. Information based adaptive robotic exploration. In *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems* (2002), vol. 1, IEEE, pp. 540–545.

[8] Caceres, R., and Iftode, L. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE Journal on Selected Areas in Communications 13*, 5 (1995), 850–857.

[9] Casetti, C., Gerla, M., Mascolo, S., Sanadidi, M. Y., and Wang, R. Tcp westwood: end-to-end congestion control for wired/wireless networks. *Wireless Networks 8*, 5 (Sept. 2002), 467–479.

[10] Castellanos, J., Montiel, J., Neira, J., and Tardos, J. The spmap: a probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation 15*, 5 (Oct 1999), 948–952.

[11] Cerf, V., Dalal, Y., and Sunshine, C. Specification of internet transmission control program. Tech. rep., INWG General Note, 1974.

[12] Chakraborty, N., and Sycara, K. Reconfiguration algorithms for mobile robotic networks. In *Proceedings of IEEE International Conference on Robotics and Automation* (2010), IEEE, pp. 5484–5489.

[13] Charrow, B., Kumar, V., and Michael, N. Approximate representations for multi-robot control policies that maximize mutual information. *Autonomous Robots 37*, 4 (2014), 383–400.

[14] Charrow, B., Liu, S., Kumar, V., and Michael, N. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. In *Proceedings of IEEE International Conference on Robotics and Automation* (Seattle, USA, 2015).

[15] Cover, T. M., and Thomas, J. A. *Elements of information theory.* John Wiley & Sons, 2012.

[16] Davids, A. Urban search and rescue robots: from tragedy to technology. *IEEE Intelligent Systems 17*, 2 (2002), 81–83.

[17] De Couto, D. S. J., Aguayo, D., Bicket, J., and Morris, R. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of Annual International Conference on Mobile Computing and Networking* (New York, NY, USA, 2003), MobiCom '03, ACM, pp. 134–146.

[18] De Gennaro, M., and Jadbabaie, A. Decentralized control of connectivity for multi-agent systems. In *Proceedings of IEEE Conference on Decision and Control* (Dec 2006), pp. 3628–3633.

[19] ElRakabawy, S. M., Klemm, A., and Lindemann, C. Tcp with adaptive pacing for multihop wireless networks. In *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing* (New York, NY, USA, 2005), MobiHoc '05, ACM, pp. 288–299.

[20] Eryilmaz, A., and Srikant, R. Joint congestion control, routing, and mac for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications 24*, 8 (Aug 2006), 1514–1524.

[21] Fink, J. *Communication for teams of networked robots.* PhD thesis, University of Pennsylvania, 2011.

[22] Fink, J., Michael, N., Kushleyev, A., and Kumar, V. Experimental characterization of radio signal propagation in indoor environments with application to estimation and control. In *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems* (Oct 2009), pp. 2834–2839.

[23] Fink, J., Ribeiro, A., and Kumar, V. Motion planning for robust wireless networking. In *Proceedings of IEEE International Conference on Robotics and Automation* (2012), pp. 2419–2426.

[24] Fink, J., Ribeiro, A., and Kumar, V. Robust control of mobility and communications in autonomous robot teams. *IEEE Access 1* (2013), 290–309.

[25] Floyd, S., Mahdavi, J., Podolsky, M., and Mathis, M. An extension to the selective acknowledgement (sack) option for tcp. Tech. rep., RFC Editor, 2000.

[26] Fu, Z., Zerfos, P., Luo, H., Lu, S., Zhang, L., and Gerla, M. The impact of multihop wireless channel on tcp throughput and loss. In *Proceedings of IEEE International Conference on Computer Communications* (March 2003), vol. 3, pp. 1744–1753 vol.3.

[27] Gil, S., Kumar, S., Katabi, D., and Rus, D. Adaptive communication in multi-robot systems using directionality of signal strength. *The International Journal of Robotics Research* (2015), 0278364914567793.

[28] Hoffmann, G. M., and Tomlin, C. J. Mobile sensor network control using mutual information methods and particle filters. *IEEE Transactions on Automatic Control 55*, 1 (2010), 32–47.

[29] Holz, D., Basilico, N., Amigoni, F., and Behnke, S. Evaluating the efficiency of frontier-based exploration strategies. In *Proceedings of International Symposium on Robotics and German Conference on Robotics* (June 2010), pp. 1–8.

[30] Hsieh, M. A., Cowley, A., Keller, J. F., Chaimowicz, L., Grocholsky, B., Kumar, V., Taylor, C. J., Endo, Y., Arkin, R. C., Jung, B., et al. Adaptive teams of autonomous aerial and ground robots for situational awareness. *Journal of Field Robotics 24*, 11-12 (2007), 991–1014.

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

[32] Hsieh, M. A., Cowley, A., Kumar, V., and Taylor, C. J. Towards the deployment of a mobile robot network with end-to-end performance guarantees. In *Proceedings of IEEE International Conference on Robotics and Automation* (2006), pp. 2085–2091.

[33] Jadbabaie, A., Lin, J., and Morse, A. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control 48*, 6 (June 2003), 988–1001.

[34] Ji, M., and Egerstedt, M. Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics 23*, 4 (Aug 2007), 693–703.

[35] Johnson, D., and Maltz, D. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, T. Imielinski and H. Korth, Eds., vol. 353 of *The Kluwer International Series in Engineering and Computer Science*. Springer US, 1996, pp. 153–181.

[36] Julian, B. J., Karaman, S., and Rus, D. On mutual information-based control of range sensing robots for mapping applications. *The International Journal of Robotics Research* (2014), 0278364914526288.

[37] Kantor, G., Singh, S., Peterson, R., Rus, D., Das, A., Kumar, V., Pereira, G., and Spletzer, J. Distributed search and rescue with robot and sensor teams. In *Field and Service Robotics*, S. Yuta, H. Asama, E. Prassler, T. Tsubouchi, and S. Thrun, Eds., vol. 24 of *Springer Tracts in Advanced Robotics*. Springer Berlin Heidelberg, 2006, pp. 529–538.

[38] Keshav, S., and Morgan, S. P. Smart retransmission: Performance with overload and random losses. In *Proceedings of IEEE International Conference on Computer Communications* (1997), vol. 3, IEEE, pp. 1131–1138.

[39] Kim, Y., and Mesbahi, M. On maximizing the second smallest eigenvalue of a state-dependent graph laplacian. *IEEE Transactions on Automatic Control 51*, 1 (Jan 2006), 116–120.

[40] Kollar, T., and Roy, N. Efficient optimization of information-theoretic exploration in slam. In *Proceedings of the National Conference on Artificial Intelligence* (2008), vol. 8, pp. 1369–1375.

[41] Kretzschmar, H., and Stachniss, C. Information-theoretic compression of pose graphs for laser-based slam. *The International Journal of Robotics Research 31*, 11 (2012), 1219–1230.

[42] Kuffner, J., and LaValle, S. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings of IEEE International Conference on Robotics and Automation* (2000), vol. 2, pp. 995–1001 vol.2.

[43] Leonard, J. J., and Durrant-Whyte, H. F. Simultaneous map building and localization for an autonomous mobile robot. In *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems* (1991), Ieee, pp. 1442–1447.

[44] Lin, M. C., Manocha, U. D., and Cohen, J. Collision detection: Algorithms and applications, 1996.

[45] Lin, X., Shroff, N., and Srikant, R. A tutorial on cross-layer optimization in wireless networks. *IEEE Journal on Selected Areas in Communications 24*, 8 (Aug 2006), 1452–1463.

[46] Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. Applications of second-order cone programming. *Linear Algebra and its Applications 284*, 1–3 (1998), 193 – 228.

[47] Lundgren, H., Nordstrom, E., and Tschudin, C. The gray zone problem in ieee 802.11b based ad hoc networks. *SIGMOBILE Mobile Computing and Communications Review 6*, 3 (June 2002), 104–105.

[48] Michael, N., Fink, J., and Kumar, V. Experimental testbed for large multirobot teams. *IEEE Robotics Automation Magazine 15*, 1 (2008), 53–61.

[49] Michael, N., Zavlanos, M., Kumar, V., and Pappas, G. Maintaining connectivity in mobile robot networks. In *Experimental Robotics*, O. Khatib, V. Kumar, and G. Pappas, Eds., vol. 54 of *Springer Tracts in Advanced Robotics*. Springer Berlin Heidelberg, 2009, pp. 117–126.

[50] Mirtich, B. V-clip: Fast and robust polyhedral collision detection. *ACM Transactions On Graphics (TOG) 17*, 3 (1998), 177–208.

[51] Mostofi, Y. Communication-aware motion planning in fading environments. In *Proceedings of IEEE International Conference on Robotics and Automation* (May 2008), pp. 3169–3174.

[52] Mostofi, Y., Gonzalez-Ruiz, A., Gaffarkhah, A., and Li, D. Characterization and modeling of wireless channels for networked robotic and control systems - a comprehensive overview. In *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems* (Oct 2009), pp. 4849–4854.

[53] Multi-robot system laboratory. `http://www.kumarrobotics.org/`, 2015.

[54] Nagle, J. Congestion control in ip/tcp internetworks. `http://tools.ietf.org/html/rfc896`, Jan 1984.

[55] Notarstefano, G., Savla, K., Bullo, F., and Jadbabaie, A. Maintaining limited-range connectivity among second-order agents. In *Proceedings of the American Control Conference* (June 2006), pp. 6 pp.–.

[56] Olfati-Saber, R., Fax, J., and Murray, R. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE 95*, 1 (Jan 2007), 215–233.

[57] Olson, E. B. *Robust and efficient robotic mapping.* PhD thesis, Massachusetts Institute of Technology, 2008.

[58] Parsa, C., and Garcia-Luna-Aceves, J. Improving tcp congestion control over internets with heterogeneous transmission media. In *Proceedings of International Conference on Network Protocols* (1999), IEEE, pp. 213–221.

[59] Pentikousis, K. Tcp in wired-cum-wireless environments. *IEEE Communications Surveys Tutorials 3*, 4 (2000), 2–14.

[60] Perkins, C., and Royer, E. Ad-hoc on-demand distance vector routing. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications* (Feb 1999), pp. 90–100.

[61] Principe, J. C. *Information theoretic learning: Renyi's entropy and kernel perspectives.* Springer Science & Business Media, 2010.

[62] Protocol, U. D. Rfc 768 j. postel isi 28 august 1980. *Isi* (1980).

[63] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. Ros: an open-source robot operating system. In *ICRA workshop on open source software* (2009), vol. 3, p. 5.

[64] Rappaport, T. S., et al. *Wireless communications: principles and practice*, vol. 2. Prentice Hall PTR, 1996.

[65] Ren, W., and Beard, R. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control 50*, 5 (May 2005), 655–661.

[66] Ribeiro, A., Giannakis, G., Luo, Z.-Q., and Sidiropoulos, N. Modelling and optimization of stochastic routing for wireless multi-hop networks. In *Proceedings of IEEE International Conference on Computer Communications* (May 2007), pp. 1748–1756.

[67] Ribeiro, A., Sidiropoulos, N., and Giannakis, G. Optimal distributed stochastic routing algorithms for wireless multihop networks. *IEEE Transactions on Wireless Communications 7*, 11 (November 2008), 4261–4272.

[68] Rimon, E., and Koditschek, D. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics 11*, 4 (1990), 412 – 442.

[69] Rimon, E., and Koditschek, D. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation 8*, 5 (Oct 1992), 501–518.

[70] Schuresko, M., and Cortés, J. Distributed motion constraints for algebraic connectivity of robotic networks. *Journal of Intelligent and Robotic Systems 56*, 1-2 (2009), 99–126.

[71] Se, S., Se, S., Lowe, D., and Little, J. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research 21* (2002), 735–758.

[72] Sim, R., and Roy, N. Global a-optimal robot exploration in slam. In *Proceedings of IEEE International Conference on Robotics and Automation* (April 2005), pp. 661–666.

[73] Sinha, P., Nandagopal, T., Venkitaraman, N., Sivakumar, R., and Bharghavan, V. Wtcp: A reliable transport protocol for wireless wide-area networks. *Wireless Networks 8*, 2/3 (2002), 301–316.

[74] Spanos, D., and Murray, R. Motion planning with wireless network constraints. In *Proceedings of the American Control Conference* (June 2005), pp. 87–92.

[75] Stachniss, C., Grisetti, G., and Burgard, W. Information gain-based exploration using rao-blackwellized particle filters. In *Proceedings of IEEE Robotics: Science and Systems* (2005), vol. 2, pp. 65–72.

[76] Stephan, J., Charrow, B., Fink, J., Michael, N., and Ribeiro, A. Simultaneous communication-aware localization and mapping with an autonomous team. *IEEE Transactions on Robotics (In Preparation)* (2016).

[77] Stephan, J., Fink, J., Charrow, B., Ribeiro, A., and Kumar, V. Robust routing and multi-confirmation transmission protocol for connectivity management of mobile robotic teams. In *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems* (Sept 2014), pp. 3753–3760.

[78] Stephan, J., Fink, J., Kumar, V., and Ribeiro, A. Concurrent control of mobility and communication in multi-robot systems. *IEEE Transactions on Robotics (Submitted)* (2015).

[79] Stephan, J., Fink, J., and Ribeiro, A. System architectures for communication-aware multi-robot navigation. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (Submitted)* (2016).

[80] Stevens, W. R., Allman, M., and Paxson, V. Tcp congestion control. *Consultant* (1999).

[81] Stump, E., Jadbabaie, A., and Kumar, V. Connectivity management in mobile robot teams. In *Proceedings of IEEE International Conference on Robotics and Automation* (May 2008), pp. 1525–1530.

[82] Tassiulas, L., and Ephremides, A. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control 37*, 12 (Dec 1992), 1936–1948.

[83] Tekdas, O., Yang, W., and Isler, V. Robotic routers: Algorithms and implementation. *The International Journal of Robotics Research* (2009).

[84] Thrun, S., Burgard, W., and Fox, D. *Probabilistic robotics*. MIT press, 2005.

[85] Tsaoussidis, V., and Badr, H. Tcp-probing: towards an error control schema with energy and throughput performance gains. In *Proceedings of International Conference on Network Protocols* (2000), IEEE, pp. 12–21.

[86] Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., and Shochet, O. Novel type of phase transition in a system of self-driven particles. *Physical Review Letters 75* (Aug 1995), 1226–1229.

[87] Visser, A., and Slamet, B. Balancing the information gain against the movement cost for multi-robot frontier exploration. In *European Robotics Symposium 2008*, H. Bruyninckx, L. Přeučil, and M. Kulich, Eds., vol. 44 of *Springer Tracts in Advanced Robotics*. Springer Berlin Heidelberg, 2008, pp. 43–52.

[88] Willow garage. `http://www.willowgarage.com/`, 2015.

[89] Wu, Y., Ribeiro, A., and Giannakis, G. Robust routing in wireless multi-hop networks. In *Proceedings of Annual Conference on Information Sciences and Systems* (2007), pp. 637–642.

[90] Xbee 802.15.4. `http://www.digi.com/products/xbee-rf-solutions/modules/xbee-series1-module#specifications`, Aug 2015.

[91] Xylomenos, G., and Polyzos, G. Tcp and udp performance over a wireless lan. In *Proceedings of IEEE International Conference on Computer Communications* (1999), vol. 2, pp. 439–446 vol.2.

[92] Yamauchi, B. A frontier-based approach for autonomous exploration. In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation* (1997), IEEE, pp. 146–151.

[93] Yan, Y., and Mostofi, Y. Robotic router formation in realistic communication environments. *IEEE Transactions on Robotics 28*, 4 (Aug 2012), 810–827.

[94] Yi, Y., and Shakkottai, S. Hop-by-hop congestion control over a wireless multi-hop network. *IEEE/ACM Transactions on Networking 15*, 1 (Feb 2007), 133–144.

[95] Zavlanos, M., and Pappas, G. Potential fields for maintaining connectivity of mobile networks. *IEEE Transactions on Robotics 23*, 4 (Aug 2007), 812–816.

[96] Zavlanos, M., and Pappas, G. Distributed connectivity control of mobile networks. *IEEE Transactions on Robotics 24*, 6 (Dec 2008), 1416–1428.

[97] Zavlanos, M., Ribeiro, A., and Pappas, G. Mobility and routing control in networks of robots. In *Proceedings of IEEE Conference on Decision and Control* (2010), pp. 7545–7550.

[98] Zavlanos, M., Ribeiro, A., and Pappas, G. Network integrity in mobile robotic networks. *IEEE Transactions on Automatic Control 58*, 1 (Jan 2013), 3–18.

[99] Zimmermann, H. Osi reference model–the iso model of architecture for open systems interconnection. *IEEE Transactions on Communications 28*, 4 (1980), 425–432.