Hybrid Decentralized Control System for Communication Aware Mobile Robotic Teams

James Stephan, Jonathan Fink, Ben Charrow, and Alejandro Ribeiro

Abstract— **Enabling a team of robots to self-organize into a multi-hop ad hoc network as it simultaneously completes a task requires a system architecture that controls both the motion of each robot and their communication variables. In this paper, we consider this objective and propose a hybrid architecture composed of both centralized and decentralized components. This novel architecture utilizes the strengths of each component while limiting the drawbacks. The resulting system is therefore able to operate in more complex environments than decentralized systems, requiring less coordination overhead than centralized systems. We demonstrate through simulation that our hybrid system has the ability to successfully complete a task while navigating complex environments, simultaneously avoiding local minima commonly encountered in decentralized systems. Furthermore, we demonstrate through experimentation the ability of our hybrid system to achieve equal, if not greater, end-to-end date rates in comparison to centralized systems with lower coordination overhead.**

I. Introduction

Autonomous mobile robotic teams are steadily becoming viable options for various tasks, such as surveillance, reconnaissance, and search and rescue. This increase in viability is due to the recent advances in team behavior [1]–[3] and multi-hop wireless network communication [4]–[8]. For this trend to continue, such autonomous robotic teams need to preserve communication while successfully navigating increasingly complex environments. In certain scenarios, preserving communication is as important, or more important, than task completion. For instance, tasks such as constant threat tracking for security or search and rescue in rapidly deteriorating circumstances require transmitting information in a timely manner. For both tasks, offloading information at specific times, rather than throughout the task, could render the task futile. In threat tracking, for example, a delay in receiving information might allow the threat to escape. Therefore, communication requirements must be considered for the entirety of the task, not simply at specific times, as in Rendezvous. Additionally, the computation and communication overhead requirements of the team must efficiently scale with the number of robots on the team so as to allow increased team sizes.

Many systems have been proposed that seek to solve the problem of motion control of mobile robots while preserving communication. These systems generally approach the problem in one of two ways, either as a decentralized optimization problem, as in [9]–[12], or as a centralized coordination problem, as in [7], [13], [14]. These approaches have been shown to operate effectively in their respective situations, but their scope is limited. For example, in the previous work done by Zavlanos et al [9], a decentralized system that controls inter-robot communication as well as robot motion was proposed. In this system, Zavlanos et al successfully isolate communication control from motion control. They define the communication variables as the solution to a discrete-time optimization problem. Simultaneously, they implement the motion control portion of their system as a continuous time gradient descent on an artificial potential function. This leads to a system that is able to complete tasks successfully while preserving a measurable level of network connectivity in a completely decentralized fashion. As this system is decentralized, it is able to scale efficiently with the number of robots; however, it still suffers from the same shortfalls as most other local controllers, namely, local minima. As each robot only possesses locally available information and sensing capabilities, these local minima appear as a global minimum.

Typically, the response to avoiding local minima is to develop a centralized system that can take advantage of global knowledge [7], [13], [14]. Such global knowledge allows the system to forgo short term benefits in order to achieve a global minimum. This allows the system to determine trajectories that enable the team to complete a given task while maintaining connectivity in the presence of local minima. However, as the number of robots on the team increases, a greater amount of communication and coordination between the robots is needed. Accordingly, while a centralized system may resolve the issue of local minima, its cost is the reduction in the ability of the system to scale efficiently.

The main contribution of this work is to combine the two disparate approaches described above, local and centralized, into a hybrid system. In particular, we use a centralized system to first determine feasible trajectories for each robot on the team. This information is then disseminated to the robots and used by their local controllers to guide them through the environment. This hybrid system provides a team of robots with the ability to move through environments that are more complex than those navigated by a local controller. This is done while avoiding the communication and coordination overhead experienced by centralized controllers.

In section II, we formalize the problem of motion control

J. Stephan, B. Charrow and A. Ribeiro are with the University of Pennsylvania, Philadelphia, PA, USA. jstephan, bcharrow, and aribeiro at seas.upenn.edu.

J. Fink is with the US Army Research Laboratory, Adelphi, MD, USA. jonathan.r.fink3.civ at mail.mil.

This research is funded by ARL MAST-CTA under Grant W911NF-08-2-0004 and the AFOSR MURI under Grant FA9550-10-1-0567.

while maintaining network connectivity. In section III, we propose a hybrid system that will combine the reactive nature of a local controller with the global performance of a centralized planner. In section IV, we present simulations that demonstrate the benefits of our hybrid approach over existing decentralized systems. Finally, in section V, we present experimental results to further prove the efficacy and robustness of our system compared to centralized approaches.

II. Problem Statement

We begin by restating the problem formulated in [7]. Consider a team of N robots, and indicate their positions as $x_i \in \mathbb{R}^2$ for $i = 1, \ldots, N$. We assume that each robot is kinematic and fully controllable to allow us to consider a simple control model of the form $u_i(t) = \dot{x}_i(t)$ where $u_i(t)$ is the control input for robot i. Also consider M access points, APs, with positions $x_i \in \mathbb{R}^2$ for $i = N+1, \ldots, N+M$. Next we define the vectors $\mathbf{x} = (x_1, \ldots, x_{N+M}) \in \mathbb{R}^{2(N+M)}$, and $\dot{\mathbf{x}} = (\dot{x}_1, \ldots, \dot{x}_{N+M}) \in \mathbb{R}^{2(N+M)}$. The team's task is given as a scalar convex potential function $\Gamma(\mathbf{x}) : \mathbb{R}^{2(N+M)} \rightarrow \mathbb{R}$. When the team achieves a configuration $\mathbf{x}^*$ for which $\Gamma(\mathbf{x}^*) = \Gamma_{\min}$ we say that team has successfully completed the task. Combining the single integrator model of the robots with the potential function we can write the following optimization problem:

$$\begin{aligned} \min_{\dot{\mathbf{x}}(t), t \in [t_i, t_f]} \quad & \Gamma\left(\mathbf{x}\left(t_f\right)\right) \\ \text{s.t.} \quad & \mathbf{x}(t) = \mathbf{x}(t_i) + \int_0^t \dot{\mathbf{x}}(s) ds. \end{aligned} \tag{1}$$

Additionally, as the robots move through the environment they must also maintain network integrity, which we define as simultaneously sustaining K data flows. We enumerate each data flow by $k = 1, \ldots, K$. For a given data flow k there can be multiple sources and destinations. For flow k we define the set of sources and destinations as src(k) and dest(k), respectively. For node i, and flow k we define the variable a_i^k as the rate at which node i can communicate with any node in the set dest(k). Likewise we define $a_{i,min}^k$ as the minimum desired rate that node i can communicate with any node in dest(k).

Next we model the inter-agent communication though a point-to-point rate function, $R(\mathbf{x}) = R(x_i, x_j) : \mathbb{R}^4 \rightarrow [0, 1]$. This function indicates the rate at which node i can send data to node j, located at positions x_i and x_j, normalized by the maximum inter-robot nominal communication rate. Due to direct communication between a source and destination not always being feasible for a given flow, the nodes must self organize into a multi-hop wireless network to relay the packets. To model this, we introduce K routing solutions, $\alpha^k(\mathbf{x}) \in \mathbb{R}^{(N+M)\times(N+M)}$. These routing solutions are used to instruct each node as to which percent of incoming data for flow k to send to each of its neighbors. Specifically, $0 \leq \alpha_{ij}^k(\mathbf{x}) \leq 1$, indicates the percentage of incoming data for flow k that node i should send to node j. To maintain queue stability at each node, we also require that the amount of incoming packets not destined for node i should never exceed the amount of outgoing packets for node i. Also since α_{ij}^k represents a percentage of time we require that $\sum_{j,k} \alpha_{ij}^k(\mathbf{x}) \leq 1$ for all i. These routing variables, $\alpha_{ij}^k(\mathbf{x})$ combined with the point-to-point model, $R(x_i, x_j)$, allow us to determine the rate at which agent i can add data to flow k.

$$a_i^k(\boldsymbol{\alpha}, \mathbf{x}) = \sum_{j=0}^{N+M} \alpha_{ij}^k R(\mathbf{x}) - \sum_{j=0, j \notin \text{dest}(k)}^{N+M} \alpha_{ji}^k R(\mathbf{x}). \tag{2}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{(N+M)\times(N+M)\times K}$ and contains all K routing solutions. Using (2) we construct a series of flow integrity constraints,

$$a_i^k(\boldsymbol{\alpha}, \mathbf{x}) \geq a_{i,min}^k, \qquad \forall i, k. \tag{3}$$

When (3) is satisfied for every i and k, given $\mathbf{x}$ and $\boldsymbol{\alpha}$, we say that network integrity is preserved.

Since we require that network integrity be maintained for the duration of the task we modify the control problem in (1) to include the flow constraints in (3). To solve this problem requires a system that attempts to control both robot motion and packet routing in order to satisfy the constraints in (1) and (3) while minimizing $\Gamma(\mathbf{x})$. It is not uncommon for the task potential function minimization and the network preservation constraint to have conflicting requirements. Due to this conflict we seek to find a joint solution to

$$\begin{aligned} \min_{\dot{\mathbf{x}}(t), \boldsymbol{\alpha}(t), t \in [t_i, t_f]} \quad & \Gamma\left(\mathbf{x}\left(t_f\right)\right) \\ \text{s.t.} \quad & \mathbf{x}(t) = \mathbf{x}(t_i) + \int_0^t \dot{\mathbf{x}}(s) ds \\ & a_i^k(\boldsymbol{\alpha}(t), \mathbf{x}(t)) \geq a_{i,min}^k. \end{aligned} \tag{4}$$

By solving for position control and network preservation simultaneously, we know that if there is a solution with $\Gamma(\mathbf{x})$ minimized then it is achieved while preserving network integrity for the entire duration of task execution. The problem formulation in (4) contains the motion control and path planning complications seen in many multi-robot problems, but the introduction of the communication requirements dramatically increases the complexity of the problem.

III. Hybrid Controller

The complexity associated with both computing and executing the solution to (4) has led to two distinct classes of systems, one class that operates as a completely decentralized system, as in [9]–[12], and one class that operates as a completely centralized system, as in [7], [13], [15]. Both approaches have distinct benefits and drawbacks when compared to the other. The centralized systems have the ability to exploit global knowledge in order to find solutions that avoid local minima. Unfortunately, the reliance on global knowledge requires a large amount of coordination and data transfer between robots, which only grows as the number of robots increases. This limits the number of robots that are allowed on the team and thus the tasks that can be completed. The decentralized systems, on the other hand, only use locally available information to operate and thus scale more favorably with the number of robots. By only using locally

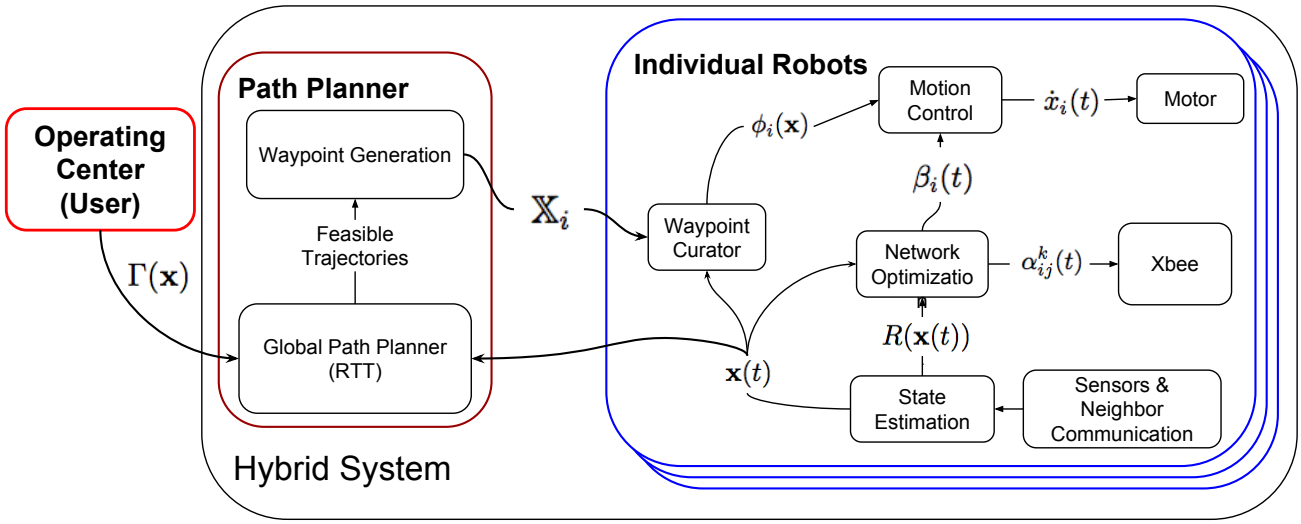


Fig. 1: Hybrid system with a slow centralized component responsible for path planning and a fast decentralized component responsible of motion control and routing solutions or each robot.

available information, the system is prone to becoming trapped in local minima. Therefore, the decentralized systems are limited to operating in relatively simple environments.

The ability of a team of robots to operate in complex environments while avoiding local minima without a heavy reliance on a centralized algorithm is the motivating case of this paper. The creation of a hybrid architecture allows a system to leverage the desirable properties of both a centralized and a decentralized system while avoiding their pitfalls. The components of this architecture build upon the previous work in [7], [9]. The hybrid architecture that we propose decomposes the problem in (4) into a two stage feedback process, with the output of one stage serving as input for the other stage.

A. Global Path Planner

Stage 1 is responsible for global path planning and is performed by only one node with the results disseminated to the other nodes. Where stage 1 executes is irrelevant, but since it is computationally expensive and the information needs to be disseminated to all the other nodes, an AP is preferable to a robot. The input to this stage is the user generated task potential function, $\Gamma(\mathbf{x})$, and the starting or current positions of the robots, $\mathbf{x}(t_i)$. This stage then computes trajectories for each robot in such a way that the network integrity constraint given by (3) is satisfied throughout robot transit. From these trajectories a set of waypoints, $\mathbb{X}_i$, is generated for each robot. These waypoints are transmitted to each robot and serve as the input to stage 2. The creation of $\mathbb{X}_i$ is explained later in section III-C.

In order to find trajectories that satisfy (4), we borrow the path planning system from Fink et al [7]. This system seeks to find trajectories for which (3) is guaranteed in a probabilistic manner. This modification to the network integrity constraint is due to the understanding that fading on a wireless channel is a random variable, and thus the inter-agent rates are also random variables. In Fink et al and in this paper, fading is modeled as a Gaussian random variable with zero mean and known variance. This leads (4) to become,

$$\begin{aligned} \min_{\dot{\mathbf{x}}(t),\boldsymbol{\alpha}(t),t\in[t_i,t_f]} \quad & \Gamma\left(\mathbf{x}\left(t_f\right)\right) \\ \text{s.t.} \quad & \mathbf{x}(t) = \mathbf{x}(t_i) + \int_0^t \dot{\mathbf{x}}(s)ds \\ & \mathrm{P}\left[a_i^k(\boldsymbol{\alpha}(t),\mathbf{x}(t)) \geq a_{i,min}^k\right] > \epsilon, \end{aligned} \tag{5}$$

where ϵ is the probability of network integrity being preserved. In Fink et al, it is shown that the network integrity constraint for a given formation can be formulated as a Second Order Cone Problem (SOCP). Therefore, feasible routing solutions can be efficiently determined, if they exist for the given formation, provided an $\epsilon > 0.5$ is selected. With the ability to efficiently evaluate the feasibility of a given formation, the global path planner can determine trajectories that allow the team to successfully complete the task while maintaining network integrity. To achieve this, the system implements a rapidly exploring random tree (RRT) algorithm to sample the configuration space, while using the SOCP formulation to only consider formations that satisfy the probabilistic network integrity constraint. From the resulting tree, trajectories can be obtained such that the constraints from (5) are satisfied, as well as, $\Gamma(\mathbf{x}(t_f)) = \Gamma_{min}$. More information on the specifics of the global path planner and RRTs can be found in [7].

B. Local Controller

This stage executes locally on each node in the network, including APs, and is individually responsible for motion control and packet routing. The input of the local controller for robot i is the set of waypoints that it should visit, $\mathbb{X}_i$, which is generated by stage 1. For APs, no waypoints are given since they are assumed to be fixed. The set of waypoints serve as guideposts that are used by a local controller

to drive each robot to a configuration that approximates the feasible configuration verified by the SOCP in stage 1. The local controller simultaneously determines the direction and velocity as well as the routing solution of the robot in order to reach the current waypoint while maintaining the network. This is achieved by only requiring a robot to exchange information with its immediate neighbors. This leads to a system that is able to scale with the number of robots because even as N gets very large the number of immediate neighbors does not increase very much.

The local controller that we are using comes from Zavlanos et al [9]. In that paper it is shown that by using the process of dual decomposition one can construct a local controller that is able to maneuver a team of robots to complete a simple task while preserving network integrity. This is achieved by having the robots share their information with their immediate neighbors. The channel model used by the local controller is simpler than the one in the global path planner in order to reduce computation load. With the reduction of the computational load, the local controller is able to operate at higher frequency, and thus react more quickly to changes in the environment. Specifically, the model used is the deterministic function:

$$R(x_i, x_j) = a\|x_{ij}\|^3 + b\|x_{ij}\|^2 + c\|x_{ij}\| + d. \qquad (6)$$

With $R(x_i, x_j) = 1$ when $\|x_{ij}\| < x_l$ and $R(x_i, x_j) = 0$ when $\|x_{ij}\| > x_u$, where $x_{ij} \triangleq x_i - x_j$. The resulting local controller solves the following in real time via dual descent

$$\begin{aligned} \max_{r_i, T_{ij}} \quad & \sum_{i=1}^{N} U_i(r_i) + \sum_{i=1}^{N} \sum_{j=1}^{N+M} V_{ij}(T_{ij}) \\ \text{s.t} \quad & r_i + \sum_{j=1}^{N} T_{ji} R(x_j, x_i) \leq \sum_{j=1}^{N+M} T_{ij} R(x_i, x_j), \\ & r_i \geq r_{i,min}, \quad \sum_{i=1}^{N+M} T_{ij} \leq 1, \quad \forall i \in \{1 \dots N\} \end{aligned} \qquad (7)$$

where $T_{ij} = \sum_{k=1}^{K} \alpha_{ij}^k$, and $r_{i,min} = \sum_{k=1}^{K} a_{i,min}^k$. Additionally the motion of robot i is controlled by means of a navigation function that seeks to minimize $\phi_i(\mathbf{x})$ while avoiding physical obstacles and loss of network integrity. More information on the dual decomposition and navigation function can be found in [9].

C. Integration

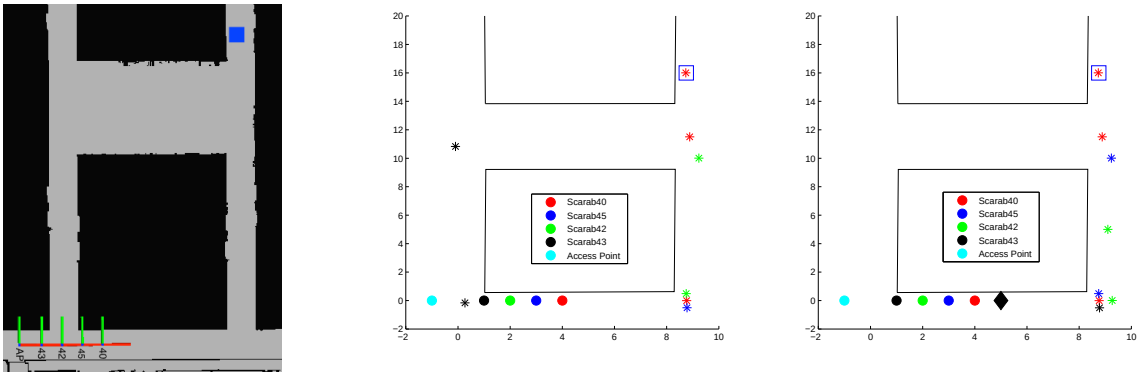
The integration of the centralized planner from Fink et al and the local controller from Zavlanos et al allows us to create a hybrid architecture that allows a team of robots to navigate more complex environments with less coordination overhead than the two systems separately. A diagram of this architecture can be seen in fig. 1. In this system stage 1, the centralized portion, runs only when a new user-defined task is given, thus implying a long time between iterations. This means that the computationally expensive and communication heavy process in stage 1 executes infrequently, possibly only once. Conversely, since stage 2 operates in real time it runs at a higher fixed frequency.

For the hybrid system to operate effectively, stage 1 must generate beneficial waypoints from the trajectories generated by the global path planner. These waypoints need to allow the local controller the freedom to locally optimize the resulting robot motion while still providing ample guidance to avoid local minima. In order to achieve this, we developed a process by which the generated paths are simulated and the robot positions are sampled at time intervals, τ_t. Such a sampling results in a set of waypoints $\mathbb{X}_i = \{x_i(\tau_t)\}_{t=1}^{T}$, with $x_i^{\tau_t} = x_i(\tau_t)$. The set of configurations are then used in stage 2 as intermediate waypoints for each robot. These waypoints are used to define the goal function, $\phi_i(\mathbf{x}(t)) = \|x_i - x_i^{\tau_t}\|$, for the local controllers. When $\phi_i(\mathbf{x}(t)) < \kappa$, t is incremented and $\phi_i(\mathbf{x})$ is updated, the robot proceeds to the next waypoint. On the last waypoint, the support robots set $\phi_i = 1$, while the lead robot keeps the last waypoint in order to maintain the objective.

Notice that as $\tau_{t+1} - \tau_t = \Delta\tau_t \to 0$ the local controller is increasingly required to follow the exact path prescribed by the global path planner. While this may prevent the local controller from becoming stuck in local minima, it foregoes the advantage of a reactive controller that is able to locally optimize the paths taken. Also, since an RRT is used to generate the paths, there is no guarantee on either optimality or smoothness, only feasibility. On the contrary, as $\Delta\tau_t \to \infty$, the system approaches the completely decentralized controller used by Zavlanos et al and is given complete freedom to optimize the paths taken. Unfortunately, the probability of becoming trapped in a local minima dramatically increases as the spacing between waypoints increases.

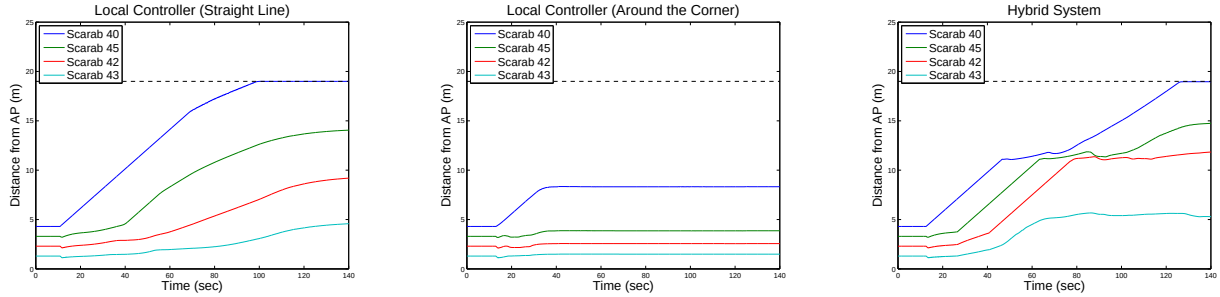
There are various ways in which the τ_t can be selected. One method involves equally spacing the τ_t in the time domain. Two other methods involve choosing τ_t such that sampling occurs after a specified robot has traveled a certain distance, or the total distance traveled by the team is above some threshold. All three of these sampling methods will produce different sets of waypoints. In this paper, we elected to implement a fixed time spacing method since it is most resilient to local minima. This is due to the distance between two consecutive waypoints for every robot being limited to the product of the sampling interval and the maximum allowed velocity. The other two methods have the possibility of one robot having to travel a very long distance between waypoints. This is analogous to the situation above where $\Delta\tau_t \to \infty$, as such the likelihood of becoming trapped in a local minima is dramatically increased. While the two distance methods are useful in certain scenarios, the robustness to local minima was paramount in our decision of sampling methods.

For the simulations in section IV and the experiments in section V, the following parameters were used. For the global path planner, $K = 1$, $\alpha_{1,min}^1 = 0.5$, $\epsilon = 0.75$. For the local controller, $x_l = 0.01$, $x_u = 0.2$, $r_{1,min} = 0.5$. Since the values chosen for the channel model in the local controller result in very small separating distance, we introduced a scale factor , $\delta = 75$. The purpose of δ is to scale the small distances used in the previous work to ones more consistent



(a) Environment that was used. (b) The waypoints used in section V-A. (c) The waypoints used in section V-B.

Fig. 2: The environment for all the tests and waypoints used in the experiments in section V. In figures (b) and (c) the circles indicate the starting location, and the stars indicate the waypoint associated with that robot. The black diamond in figure (c) is the location at which *Scarab*43 stalls in section V-B



(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.

Fig. 3: Simulation results for local and hybrid systems. For all tests the goal location is 19 meters away.

with the *Scarab* platform. The scaling factor is used in two places; first, to translate the physical positions to the system's units and second, to amplify the control law output. The entire hybrid system is implemented in the Robotic Operating System (ROS) so that it can be simulated using kinematically accurate models as well as executed on robotic platforms.

The benefits of this hybrid approach are twofold. First, by leveraging the global knowledge of a centralized planner, the team is not only able to avoid local minima but is also able to find feasible trajectories, if they exist. Second, by allowing the robots to deviate from the global plan in between waypoints, the system is able to achieve higher successful transmission rates. This is due to the robots reacting to the changes in the configuration and dynamically adjusting both positions and routing probabilities to optimize the input rates r_i.

IV. Simulations

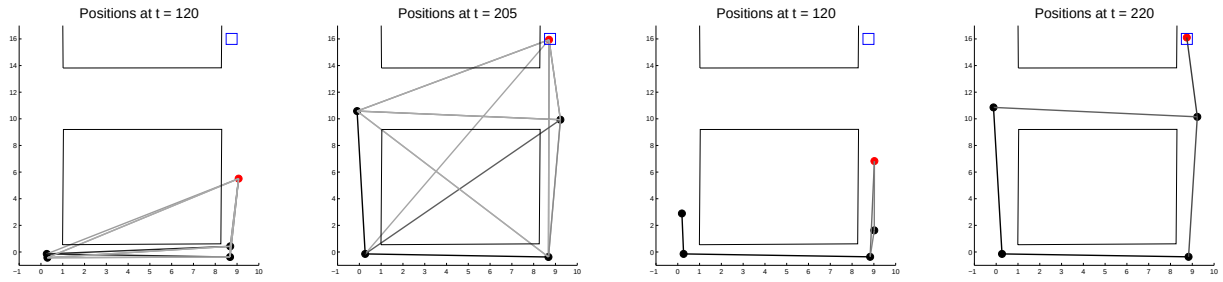
In this section we present a series of simulations that show how the local controller from [9] performs in different environments. We also show how our hybrid system allows the team to complete objectives that are unachievable when only the local controller is used.

A. Test Environment

The test environment used in both this section and section V is the 5^{th} floor of the Graduate Research Wing at the University of Pennsylvania, shown in Fig. 2a. The number of robots, 4, and their starting positions were the same for all of the simulations and experiments. The access point is located in the lower left corner, with the robots spaced 1 meter apart going up the hallway, as indicated by the green and red axis in the image. The order of the robots in increasing distance from the access point are *Scarab*43, *Scarab*42, *Scarab*45, *Scarab*40, with *Scarab*40 being the lead robot. The blue square in the upper right corner indicates the goal location used in the majority of the tests.

B. Pure Local System

In the first simulation, *Scarab*40 is provided a goal that is 19 meters straight down the hallway. The distance of each robot from the access point is plotted in fig. 3a. The robots begin moving 10 seconds into the simulation and the lead *Scarab* reaches the goal in less than 90 seconds. Since no goal is given to the supporting robots, they move in order to maximize the rate of the lead robot, r_1.



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

Fig. 4: These figures show a series of formations and the resulting routing probabilities experienced during the experiments in section V-A. Figures (a)-(b) correspond to the centralized system experiments and figures (c)-(d) correspond to the hybrid system experiments. The darkness of the lines connecting the robots indicate the routing probabilities used for that link.

In the next simulation, the goal is still 19 meters away, but this time in order to reach the goal the robot needs to turn a corner. The walls and corner prevent the lead robot from heading directly towards the goal and therefore introducing a local minimum. This local minimum is enough to prevent the lead robot from reaching the goal. This can be seen in fig. 3b by the plateauing of the blue line well below the goal distance. The local minimum is due to the forces from maximizing network integrity and obstacle avoidance canceling out the force of the goal function. The added force caused by the team's attempt to maximize network integrity only increases the number of local minima. As with most local controllers, when the goal changes, so does the local minima. This, unfortunately, introduces a dependence for task completion on the selection of goals, which might not be well understood in some environments. This simulation highlights the limitations of a purely local system and suggests the need to design a system that is more robust to local minima.

C. Hybrid System

Since the purely local system is unable to successfully reach the goal around the corner, the hybrid system is used in the next simulation. The global planner validated that the goal location for the lead robot was in fact achievable and moreover, network integrity could be preserved throughout execution. The results of the simulation are in fig. 3c, where again the distance of each robot from the access point is plotted. Notice that the team is now able to successfully navigate the complex environment and reach the goal. In this simulation, the lead robot is able to reach the goal in roughly the same amount of time as the straight ahead case shown in Fig. 3a. This shows that in the scenarios where local minima prevent the purely local system from achieving the goal, a hybrid approach is able to pull the team out of such minima and eventually reach the goal.

V. Experimental Validation

This section focuses on the series of experiments that first compare the performance of the hybrid system to the centralized system, then highlight the benefits of a hybrid system over a centralized system when a robot has a temporary motor failure.

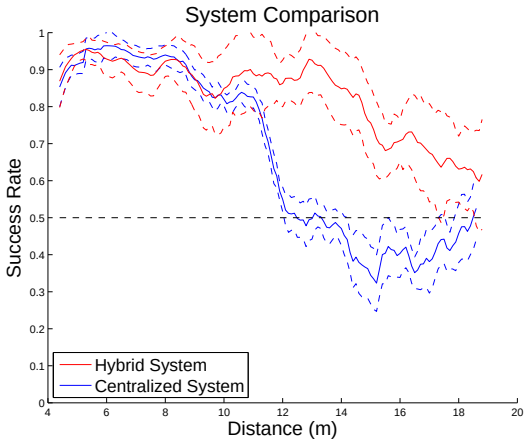
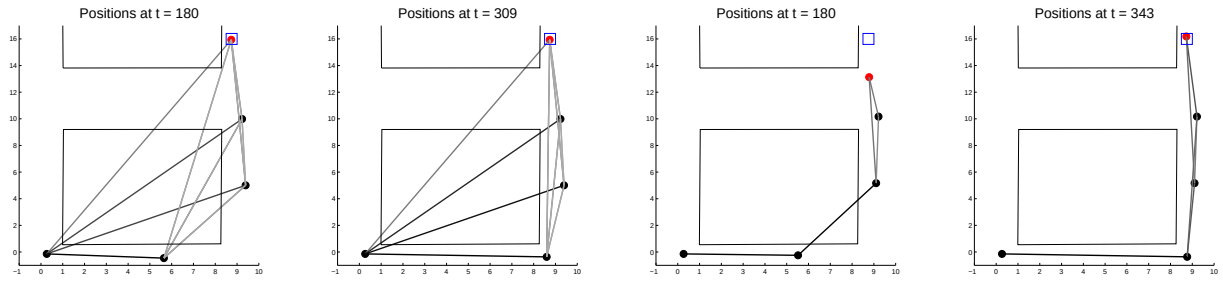


Fig. 5: Experimental results for centralized and hybrid systems. The solid line is the average performance and the dashed colored lines are +/- 1 σ bounds. The black dashed line is the minimum input data rate for the lead robot.

For this paper we use *Scarab*s [16], a custom built robot designed at the University of Pennsylvania, as our robotic platform. For more information on the exact configuration of the *Scarab* platform used in the experiments see [15]. The only change made to the configuration in [15] is that each Xbee is configured to transmit at -10 dBm to limit the signal propagation between robots.

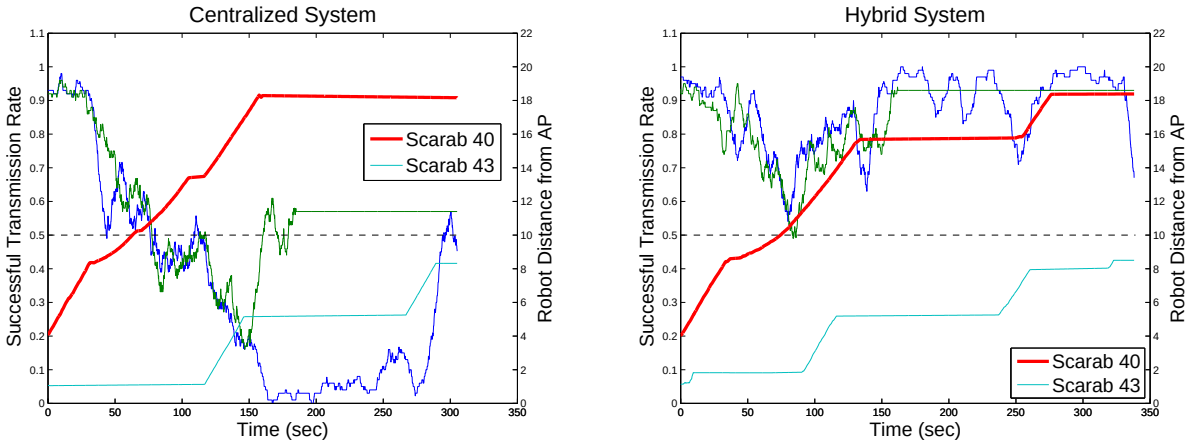
A. System Performance

The performance of the hybrid and centralized systems is shown in fig. 5. We measure the performance of the system by computing the percentage of messages, sent by *Scarab*40, that are successfully received at the AP. Both systems are tasked with maintaining the same level of successful packet transmission, $r_{1,min}$, for *Scarab*40. In these experiments the same set of waypoints were used for both the centralized and the hybrid systems to remove any planning bias, shown in fig. 2b. Also 10 sets of data were collected for both systems.



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

Fig. 6: These figures show a series of formations and the resulting routing probabilities experienced during the experiments in section V-B. Figures (a)-(b) correspond to the centralized system experiments and figures (c)-(d) correspond to the hybrid system experiments. Figures (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.

Fig. 7: Experimental results highlighting the hybrid systems ability to dynamically adjust to motor failures. The green and blue lines indicate the experienced data rate for tests where the robot's motor does and does not fail.

This allows the random perturbations caused by fading and the environment to be smoothed out.

The first item of import is that the hybrid system always maintains network integrity, $r_1 \geq 0.5$, throughout the experiment. Conversely, the centralized system fails to preserve network integrity during portions of the experiment. This is shown by its rate, the blue line, falling below the dotted black line. Another important feature to notice is that the hybrid system consistently outperforms the centralized system. This is due to the hybrid system optimizing the robot motion in between the waypoints while the centralized system is only executing a prescribed path for each robot. The final item to note is the tight σ bounds, as indicated by the colored dashed lines, in the centralized system when compared to the hybrid system. This can be attributed to two causes. First, the hybrid system locally optimizes the paths, thus increasing the variability of the intermediate formations. Second, the formulation of the routing optimization problem is different between the two systems. Specifically the centralized system considers both the fading and expected value of the channel rates, while the hybrid system only considers the later. This can be seen in fig. 4 where snapshots in time of the formation and routing probabilities are shown for both the centralized and the hybrid systems. Even though the formations are similar notice that the centralized system prioritizes link diversity over data rate. This results in the tighter σ bounds at the expense of the mean data rate.

B. Robustness to Failures

The design of the hybrid system leverages the reactive nature of a local planner to create a system that is robust to various types of failures. This can be seen in a scenario where there is an issue with the motion of the robots. The experiments in this section highlight such a scenario.

The two systems are run, as in the previous experiments, but with a different set of waypoints, show in fig. 2c. As with the previous experiments the same waypoints are used for both systems to remove planning bias. Also in these experiments one of the support robots, *Scarab*43, stalls for 2 minutes at a specified point in the path. Even though

the same waypoints are used for the centralized and the hybrid system the results are very different. The performance of the centralized system and hybrid system are plotted in fig. 7a and fig. 7b, respectively. On the left y-axis, the blue and green lines represent the average successful packet transmission for the lead robot in an experiment where the *Scarab*43 stalls and an experiment where there is no stalling. Also a dashed line representing the minimum rate, $r_{1,min} = 0.5$, is included. On the right y-axis, the distance of the lead and stalled robots from the access point are plotted. Notice that when the support scarab stalls in fig. 7a, at $t = 140$, the lead robot continues to the goal, as indicate by the red line reaching its maximum value at $t = 160$. This is due to the centralized system executing the path as prescribed with no feedback. The result is a drop in the performance of the transmission success rate since the support robot is not in the expected location. This is evident by the blue and green line tracking each other for the initial part of the experiment. When *Scarab*43 stalls the two line diverge. The green line increases because in that experiment *Scarab*43 is in the correct location. Conversely, the blue line continues to decrease because the actual robot formation differs greatly from the expected robot formation, shown in fig. 6a. When the robot recovers, the success rate returns to the higher value because *Scarab*43 finally reaches its expected location.

In contrast, notice that when the support robot stalls in fig. 7b, at $t = 120$, the lead robot continues to a point but itself stalls due to network integrity beginning to suffer. This is also seen in fig. 6c. Eventually, when the support robot recovers, the lead robot resumes its motion and reaches the goal. In this set of experiments the blue and green line track during the initial part, similar to the centralized case, and again diverge when the support robots stalls. Fortunately, even though the two lines diverge they both stay above the dotted line indicating that network integrity is maintained. The performance of the hybrid system highlights the benefits of a reactive motion controller.

VI. Conclusions

We considered the problem of controlling a team of robots that seek to complete a task while maintaining network integrity. We propose a hybrid system that builds on the previous work to provide advantages beyond what a local or global controller can provide individually. The results of this paper demonstrate that by designing a hybrid system, the benefits of a decentralized system can be combined with the benefits of a global path planning system. Specifically, the hybrid system combines the lower communication overhead and reactive motion planning from the decentralized system, with the guarantee of task completion from the global system.

In future work, we plan to focus on including more complex forms of the channel reliability function, $R(x_i, x_j)$, such as using the instantaneous received signal strength indicator included in every receive packet in order to model the expected reliability of the current communication link. This work focused on the design and implementation of a hybrid system that allows a team of robots to successfully complete a task while preserving the integrity of a wireless network.

References

[1] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan 2007.

[2] M. Zavlanos and G. Pappas, "Distributed connectivity control of mobile networks," in *Decision and Control, 2007 46th IEEE Conference on*, Dec 2007, pp. 3591–3596.

[3] G. Antonelli, F. Arrichiello, F. Caccavale, and A. Marino, "Decentralized time-varying formation control for multi-robot systems," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1029–1043, 2014. [Online]. Available: http://ijr.sagepub.com/content/33/7/1029.abstract

[4] A. Ewerlid, "Reliable communication over wireless links," in *Nordic Radio Symposium 2001 (NRS 01*, 2001, pp. 3–5.

[5] A. Ribeiro, G. Giannakis, Z.-Q. Luo, and N. Sidiropoulos, "Modelling and optimization of stochastic routing for wireless multi-hop networks," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, 2007, pp. 1748–1756.

[6] J. Fink, "Communication for teams of networked robots," Ph.D. dissertation, University of Pennsylvania, 2011.

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

[8] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," 2003.

[9] M. Zavlanos, A. Ribeiro, and G. Pappas, "Network integrity in mobile robotic networks," *Automatic Control, IEEE Transactions on*, vol. 58, no. 1, pp. 3–18, Jan 2013.

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

[11] M. Schuresko and J. Cortes, "Distributed motion constraints for algebraic connectivity of robotic networks," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, 2008, pp. 5482–5487.

[12] D. Spanos and R. Murray, "Motion planning with wireless network constraints," in *American Control Conference, 2005. Proceedings of the 2005*, 2005, pp. 87–92.

[13] E. Stump, A. Jadbabaie, and V. Kumar, "Connectivity management in mobile robot teams," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 2008, pp. 1525–1530.

[14] O. Tekdas, W. Yang, and V. Isler, "Robotic routers: Algorithms and implementation," *Int. J. Rob. Res.*, vol. 29, no. 1, pp. 110–126, Jan. 2010. [Online]. Available: http://dx.doi.org/10.1177/0278364909105053

[15] J. Stephan, J. Fink, B. Charrow, A. Ribeiro, and V. Kumar, "Robust routing and multi-confirmation transmission protocol for connectivity management of mobile robotic teams," in *ROS 2014 IEEE International Conference on*, 2014.

[16] N. Michael, J. Fink, and V. Kumar, "Experimental testbed for large multirobot teams," *Robotics Automation Magazine, IEEE*, vol. 15, no. 1, pp. 53–61, 2008.