

# We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,600

Open access books available

178,000

International authors and editors

195M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index  
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?  
Contact [book.department@intechopen.com](mailto:book.department@intechopen.com)

Numbers displayed above are based on latest data collected.  
For more information visit [www.intechopen.com](http://www.intechopen.com)



## Chapter

# A Multi-Layer, Multi-Robot Control Architecture for Long-Range, Dynamic Communication Links

*John Shepard and Christopher Kitts*

## Abstract

A unified motion control architecture is presented for dynamic, long-range multi-robot communications networks, incorporating task abstraction that disassociates goals from implementation. In the task space, communication link states are specified, directly measured, and explicitly controlled yielding well-behaved task state trajectories. The control architecture uses task-level compensation to generate multi-robot formation mobility commands, and a cluster space controller transforms those formation commands to mobility commands for individual robots. The number of robots are selected to meet communications requirements and controlled through a multi-task coordination capability incorporated within the architecture. Robustness to performance commands, system configuration parameters, and external disturbances is demonstrated through a variety of simulations and experiments. These show how robots are dynamically positioned and switched into or out of operation in order to meet communications requirements.

**Keywords:** multi-robot systems, mobile robots, communication networks, robot control, collaborative robots

## 1. Introduction

Robotic systems are an integral tool in modern society, extending the capability of human operators, increasing their productivity and improving their quality of life. Multi-robot systems are able to enhance quantity-sensitive performance metrics like speed, coverage, throughput, and redundancy. In addition, their spatial diversity provides added capabilities for tasks like formation-keeping [1–3], escorting or guarding [4, 5], surveillance and feature tracking [6–9], object manipulation [10], and more specialized tasks such as automatic lighting [11] reconfigurable sparse antenna arrays [12], and minimally invasive surgery [13, 14]. New research is developing techniques for diverse task-oriented groups of robots to work together to perform broader and more sophisticated missions [15, 16]; as the mission evolves, the tasks may be preserved even though the environment, the performance objectives, and the assignment of robots may change.

In this research, we examine the task of long-range and dynamic communication link management due to its necessity as a supporting role in many applications and missions. By “long-range” we refer to communication links that require a single series of repeater stations to relay communication between remote end stations. As such, mesh network approaches such as those described in [17, 18] which provide multiple communication paths are not appropriate for the applications of interest. By “dynamic” we refer to the need for robot relays to adjust their positions to compensate for changing link conditions due to motion of the end stations, the attenuation environment, communications equipment performance, etc.; as such, deploying static relay stations, as is done in [19, 20], is not sufficient.

Model-based approaches to link management use a model of the link along with information regarding relay robot positions in order to estimate quality of service. These are typically well behaved due to smoothing simplifications [21] but can be inaccurate. In [22], a simple binary model is used, where nodes are assumed to be connected if within a fixed distance, leading to a strategy that evenly spaces robots between end-stations. More sophisticated models may be used, incorporating path models based on power, distance, and line-of-sight in order to identify a set of goal positions for the robots [23–25].

Measurement-based approaches remove model inaccuracies, allowing for improved performance and an expanded workspace [26], and are robust to complexities of the link behavior like obstructions, directional antenna patterns, and multipath effects. For example, in [27], unmanned aircraft are used to establish a relay network, with the airborne relays circling control points and measuring link gradients to relocate to optimal locations. In [28], it was found that the use of measurement-based approaches improved performance in maintaining a line of sight connection between robots. In [29], optimization techniques exploited measurements to minimize the path traveled by relay robots as they maneuver to a fixed location to support end-station communication. In [30], a centralized planner guides a multirobot team in known indoor environments to establish a multihop network, with measurements used to improve local positioning. Finally, in [31, 32], multi-robot mesh communications is achieved using a potential-function-based, decentralized control scheme and measurements of communication bit error rate.

The work presented in this paper addresses repositioning of communication relay nodes operating within a long-range, dynamic link in order to ensure that a measured Quality of Service (QoS) level, signal strength, is maintained. Service is achieved and maintained using a multi-layer control architecture. At the highest level, a task-oriented operational space control approach produces well-behaved task state compensations. These are converted to robot formation motion commands using a model-based inverse Jacobian transform. Formation compensations are then converted to individual robot drive commands to achieve the necessary level of position control. A flexible number of robots is used to support the link, with robots being switched into and out of service as necessary; to be more precise, unused robots are actually switched to a benign position-control task that maintains their location in a ready position for future use in maintaining the link. Formally, all robots, regardless of their assigned task, are controlled through a single unified position control framework that dynamically changes the transforms used to convert between control state spaces.

Each element of this control architecture is well-motivated and provides a useful innovation compared to previous work in managing dynamic long-range communication links. First, direct sensing of and control implementation in the operational task space improves application-oriented performance. Second, use of this space abstracts

the specification of desired performance from implementation details such as the number of robots used to achieve the task and the mobility characteristics of these robots, thereby preventing the redesign of the specification process when these implementation details are changed. Third, our approach flexibly engages an appropriate number of robots in the link management task, thereby conserving resources when the full suite of robots is not required. Fourth, use of the intermediate cluster space representation provides a critical layer of abstraction that reflects the geometric nature of the application, thereby making the construction of task level transforms simpler; furthermore, this control architecture provides an intuitive intermediate layer for supervisory operators when specifying and monitoring performance as well as for developers when incrementally designing, verifying and troubleshooting functionality of the multi-robot system. Fifth, the architecture unifies motion control for all robots within a single control architecture (whether the controller itself is centralized or decentralized) thereby facilitating development and performance analysis of large-scale, multi-task missions. Overall, our approach provides enhanced performance, minimizes the use of robot resources, promotes modular composition, and has been verified experimentally; results show the technique to be robust to the dynamic link environment based on moving end stations, local attenuation, and variation in the state of the communication relays.

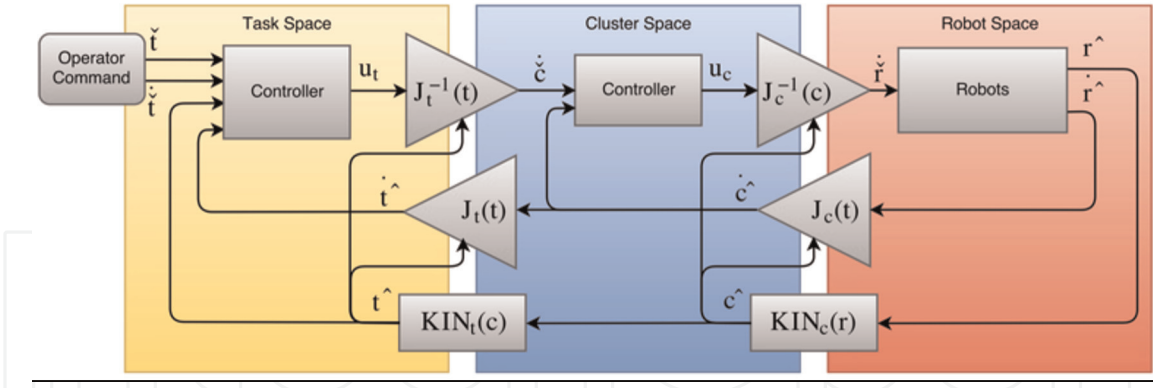
The article starts with a synopsis of the basic control architecture in Section 2, discusses extension to the task space in Section 3, integrates multiple tasks in Section 4, describes the experimental test bed in Section 5, presents results of the experiments and simulations in Sections 6, and considers future work in Section 7.

## 2. Cluster space control

The first layer of our control architecture addresses mobility control of the multi-robot cluster. While a number of techniques exist for multi-robot formation control, the Cluster Space control architecture abstracts the cluster as a virtual articulating mechanism, allowing explicit control of all system states. The interested reader should consult [33] for comparisons. The underlying goal of the Cluster Space technique is simple motion specification and control of multi-robot systems. This is accomplished by considering multiple robots as a single geometric entity. The pose of a cluster is described by its location and shape, which are related to individual robot positions through a set of kinematic transforms. For a system of  $n$  robots with  $q$  total degrees of freedom, the generalized cluster and robot pose vectors and their kinematic relationships are:

$$\vec{C} \equiv \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_q \end{bmatrix} = \begin{bmatrix} g_1(r_1 \dots r_q) \\ g_2(r_1 \dots r_q) \\ \vdots \\ g_q(r_1 \dots r_q) \end{bmatrix} \quad (1)$$

$$\vec{R} \equiv \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_q \end{bmatrix} = \begin{bmatrix} h_1(c_1 \dots c_q) \\ h_2(c_1 \dots c_q) \\ \vdots \\ h_q(c_1 \dots c_q) \end{bmatrix} \quad (2)$$



**Figure 1.**

The multilayer control architecture. Layers exist for task-space, cluster space and robot space control, with inverse Jacobian resolved rate controllers used within each space.

where  $r_i$  are robot pose states,  $c_i$  are cluster pose states,  $g_i(\dots)$  are kinematic equations, and  $h_i(\dots)$  are inverse kinematic equations [33, 34]. Cluster state velocities are linearly mapped from robot velocities using a Jacobian matrix.

These mathematical transformations are the basis for the layered cluster space control framework shown in **Figure 1**. On the right, in red, are the individual robots, which accept platform-level velocity commands. The blue region is the cluster space controller, shown in the form of a kinematic, resolved rate controller. This layer accepts cluster-level commands regarding cluster mobility and geometry; its outputs are robot velocity commands for each robot in order to achieve the cluster-level goals. Forward kinematic equations are used to compute cluster states from estimated robot states, and cluster control effort is transformed to robot velocity commands using the inverse cluster Jacobian matrix. This control technique has been implemented experimentally in a wide variety of multi-robot systems operating in land, sea, and air [4, 5, 7, 35, 36]. The yellow region of the diagram is described in the next section.

### 3. Task space control

Just as operational space [37] kinematic transforms are used to allow control of robots based on their actuator configuration, the cluster space methodology uses kinematic transforms to establish the control task in terms of multirobot formation mobility and geometry. Here, we add yet another operational space control layer for task-oriented specification of behavior. As shown in **Figure 1**, each control layer uses a resolved rate controller that provides rate commands to an inverse Jacobian function, which converts those commands to rate set-points for the next layer (full dynamic controllers have been demonstrated in other works). Using this approach, operator commands are issued and controlled at the task level, and compensation commands are then successively transformed to cluster velocity, robot velocity and finally actuator velocity set-point commands as the control architecture executes. Each successive layer acts as an inner control loop for the preceding layer.

Given this, the long-range dynamic communication link management system presented in this paper involves two interacting tasks. The first is a “communication space” task that uses a measurement-based, link-balancing control strategy to space robots along the inline dimension between the end stations (a model-based



assumption that in-line positioning is best has been made, leading to the use of a simple cross-track nulling controller for the second dimension); furthermore, a measurement-based controller varies the number of robots required by the task to minimally satisfy the desired aggregate communication quality set points. The second task is a simple position control task to maintain unused robots in a ready state. While the second task is trivial, we treat the management and interplay of these multiple distinct tasks in a formal manner.

In total, we are using a multi-layer, multi-task controller. In [24] we develop the conditions to ensure Lyapunov stability for each layer as well as the conditions for task switching. Space limitations prevents their further discussion in this work.

The following subsections demonstrate use of the layered control methodology for the communication and position control tasks. For each task, a formal definition of the layered state spaces is provided, the kinematic and Jacobian transforms used to convert between spaces are established, and controllers are provided for each layer. Both tasks assume the use of planar robots given that this was the type of robot used in the simulations and experiments described in Section VI.

### 3.1 Example task: long range communication

Consider the task of long-range communications between two exogenous nodes using mobile relays. To maintain the link quality,  $n_c$  robotic relay nodes will move to intermediate locations based on desired link characteristics.

#### 3.1.1 Spaces and states

The relevant spaces for this scenario are the individual robot space, the cluster space describing the geometry of the task-specific group of robots, and the communication task space. The robot space is defined by the pose of all robots, specified below for quantity  $n_c$  robots:

$$\vec{r} \triangleq [x_1, y_1, \theta_1, \dots, x_{n_c}, y_{n_c}, \theta_{n_c}]^T \quad (3)$$

where  $(x_i, y_i)$  is the Cartesian position and  $\theta_i$  is the orientation of robot  $i$ , assuming planar operation.

The cluster space pose vector describes the location and shape of the cluster. In this case, the separation distances  $\rho_i$  and chain angles  $\alpha_i$  define the geometry, as depicted in **Figure 2**; although many other cluster definitions are possible, this choice is convenient due to the serial nature of the communications task.

Accordingly, the cluster state vector is defined:

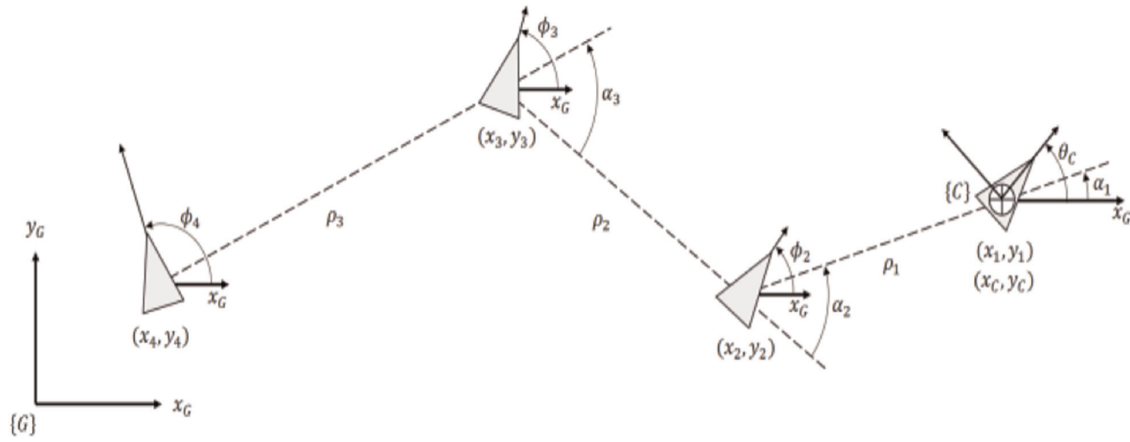
$$\vec{c} \triangleq [x_c, y_c, \theta_c, \rho_1, \alpha_1, \phi_1, \dots, \rho_{n_c-1}, \alpha_{n_c-1}, \phi_{n_c-1}]^T \quad (4)$$

In the task space, the user is interested in maintaining sufficient communication quality of service (QoS) between two end nodes, with signals being relayed as needed. Quality of service proved impractical to quantify in real time, so the system measures the link power between nodes using the received signal strength indicator (RSSI). For line-of-sight, the RSSI may be modeled as inversely proportional to the square of the distance between two points, hence:

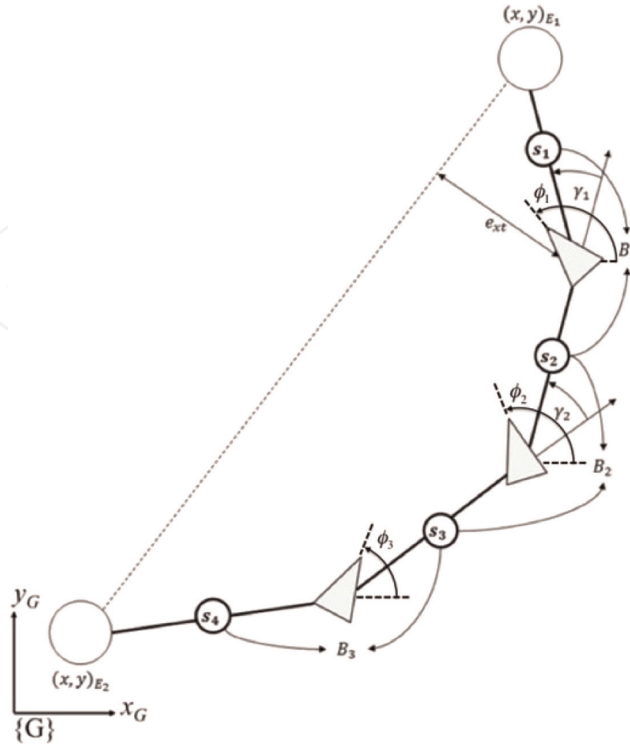
$$s_i = \frac{k}{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} = \frac{k}{\rho_i^2} \quad (5)$$

where  $k$  is a constant associated with the antenna gain. It is important to note that RSSI is measured directly but this model is used to compute the Jacobian, similar to a model-based approach for gain scheduling.

As depicted in **Figure 3**, the quality of service between the end nodes is influenced by both the crosstrack error,  $e_{xt}$ , and the angles of alignment,  $\gamma_i$ . Assuming a line of sight model, the maximum total signal strength is achieved by minimizing the crosstrack error and angles of alignment. The ratio or balance,  $B_i$ , of the link power in each segment is also important to avoid data rate bottlenecks or backup in homogeneous systems, or to allow for imbalanced transmission rates in nonhomogeneous



**Figure 2.**  
Serial chain cluster diagram for an  $i$ -hop chain.



**Figure 3.**  
Long-range link diagram for a 4-hop link performed by 3 robots.

systems. The first and last links are functions of the position of the end nodes being connected  $(x_{E_1}, y_{E_1})$ ,  $(x_{E_2}, y_{E_2})$  which are uncontrolled states of the environment. Lastly, the orientation of the robot,  $\psi_i$ , is included to fully define all degrees freedom of system. The communication “pose” vector is defined:

$$\vec{t}(x_{E_1}, y_{E_1}, x_{E_2}, y_{E_2}) \triangleq [B_1, \dots, B_{n_c}, e_{xt}, \gamma_1, \dots, \gamma_{n_c-1}, \psi_1, \dots, \psi_i]^T \quad (6)$$

Given these definitions, we define the desired states as follows. For a uniformly balanced network,  $B_i = 1$ . For minimum crosstrack error for maximum link quality,  $e_{xt} = 0$ . For aligning the robots between end points,  $\gamma_i = 0$ . In this case, the robots have a holonomic constraint and so the robot headings,  $\psi_i$ , are controlled at the platform level. Trajectory generation is a major topic of research for robotic communication networks [38] but it is not our focus in this work. These commands remain constant throughout each presented experiment, unless otherwise noted. It may be argued that more sophisticated control algorithms require less sophisticated command trajectories.

### 3.1.2 Kinematic transformation equations

Robot states are transformed into the cluster states using kinematic equations derived from formation geometry:

Cluster frame:

$$x_c \triangleq x_1 \quad y_c \triangleq y_1 \quad \theta_c \triangleq \theta_1 \quad (7)$$

Chain length:

$$\rho_i \triangleq \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \quad (8)$$

Chain angle:

$$\alpha_i \triangleq \text{atan2}(y_{i+1} - y_i, x_{i+1} - x_i) - \sum_{j=1}^{i-1} \alpha_j \quad (9)$$

Node orientation:

$$\phi_i \triangleq \theta_i \quad (10)$$

where  $\text{atan2}(\dots, \dots)$  is the two-argument function that calculates a four-quadrant arc tangent with a range of  $[\pi, -\pi]$

These cluster states are transformed into the task states using the measured link states and system geometry:

Balance:

$$B_i \triangleq \frac{s_{i+1}}{s_i} = \begin{cases} \frac{(x_{E_1} - x_{c1})^2 + (y_{E_1} - y_{c1})^2}{\rho_1^2} & \text{for } i = 1 \\ \frac{\rho_2^2}{(x_{E_2} - x_c + \rho_2 \cos(\alpha_1 + \alpha_2) + \rho_1 \cos \alpha_1)^2 + (y_{E_2} - y_c + \rho_2 \sin(\alpha_1 + \alpha_2) + \rho_1 \sin \alpha_1)^2} & \text{for } i = n_p - 1 \\ \frac{\rho_i^2}{\rho_{i+1}^2} & \text{otherwise} \end{cases} \quad (11)$$



Crosstrack error:

$$e_{xt} = \sqrt{\frac{\left((x_{E_2} - x_{E_1})(y_{E_1} - y_c) - (x_{E_1} - x_c)(y_{E_2} - y_{E_1})\right)^2}{(x_{E_2} - x_{E_1})^2 + (y_{E_2} - y_{E_1})^2}} \quad (12)$$

Angle of alignment

$$\gamma_i = \alpha_i \quad (13)$$

Orientation:

$$\psi_i = \phi_i \quad (14)$$

where  $(x_{E_1}, y_{E_1})$  and  $(x_{E_2}, y_{E_2})$  are the positions of the end stations that are being connected by the multi-robot communication system.

### 3.1.3 Jacobian matrices

The Jacobian matrices are computed from the kinematic equations to map velocities between spaces. The solution is typically lengthy and so not shown here but easily computed using (5) with (7–10) for the cluster Jacobian or with (11)–(14) for the task Jacobian.

### 3.1.4 Control design and performance

In addition to determining the kinematic transforms, control laws must also be formulated. For these experiments, simple linear controllers suffice as the testbed platforms have well-behaved dynamics, but the system architecture can accommodate any type of control algorithm within each space.

As many commercially available robotic platforms control their own local velocity, a detailed discussion of platform control is not necessary. It is important to note that if the robots are nonholonomic, orientation and global translation are coupled which precludes independent control of all degrees of freedom. For details on our testbed nonholonomic control, please see [36].

The following equations can be used to design controllers in higher spaces using traditional techniques. The transfer functions at each layer can be approximated as linear, time-invariant (LTI) with proper tuning, maintaining diagonal dominance, and avoiding singularities. General system stability and performance is discussed in Section III.C.

Cluster space response:

$$\dot{\vec{c}} = (J_c^{-1} + G_r J_c^{-1} H_c)^{-1} G_r J_c^{-1} H_c \dot{\vec{c}}_d = G_c \dot{\vec{c}}_d \quad (15)$$

Task space response:

$$\vec{t} = (J_t^{-1} sI + G_c J_t^{-1} H_t)^{-1} G_c J_t^{-1} H_t \check{t} = G_t \vec{t}_d \quad (16)$$

where  $G_x$  represents a diagonal matrix of transfer functions in space  $x$ ,  $H_x$  represents a controller in space  $x$ . The system pose is represented by  $r$  in robot space,  $c$  in cluster space, and  $t$  in task space. As subscripts, these letters associate the variable with a space. Subscript  $d$  denotes desired states.

For these particular experiments, the cluster space control law utilizes proportional feedforward and feedback, shown below, for response time and error rejection respectively:

$$\vec{u}_c = H_c \left( \dot{\vec{c}}_d, \dot{\vec{c}} \right) = K_{c_f} \dot{\vec{c}}_d + K_{c_p} \left( \dot{\vec{c}}_d - \dot{\vec{c}} \right) \quad (17)$$

where  $u_c$  denotes cluster space control effort,  $\dot{\vec{c}}_d$  denotes desired cluster velocity,  $K_{c_f}$  denotes proportional feedforward gain matrix, and  $K_{c_p}$  denotes proportional feedback gain matrix.

For these particular experiments, the communication task space uses proportional feedback control, shown below:

$$\vec{u}_t = H_t \left( \vec{t}_d, \vec{t} \right) = K_{t_p} \left( \vec{t}_d - \vec{t} \right) \quad (18)$$

where  $K_{t_p}$  is the feedback gain matrix and  $\vec{t}_d$  is the desired state. These desired states are discussed in Section III.A.1). While simplistic, these control laws yield sufficient performance in this application and our prior work with different tasks performed in land, sea, and aerial environments [4, 5, 7, 12, 39]

### 3.2 Example task: position control

In this task,  $n_p$  robots are tasked to go to and maintain a specified position. Since the task is a direct specification of individual robot positions, the task is identical to many one-robot clusters which is identical to control of individual robots. While this degenerates into trivial task, we adhere to the layered control architecture in order to provide unified control of all mission-related robots.

Space limitations prevent a complete description of this task. Given it's simplicity, key aspects of its implementation are reviewed here. To begin, the robot cluster and task pose vectors are:

$$\vec{r} \triangleq \left[ x_1, y_1, \theta_1, \dots, x_{n_p}, y_{n_p}, \theta_{n_p} \right]^T \quad (19)$$

$$\vec{c} \triangleq \left[ x_{c_1}, y_{c_1}, \theta_{c_1}, \dots, x_{c_{n_p}}, y_{c_{n_p}}, \theta_{c_{n_p}} \right]^T \quad (20)$$

$$\vec{t} \triangleq \left[ x_{t_1}, y_{t_1}, \theta_{t_1}, \dots, x_{t_{n_p}}, y_{t_{n_p}}, \theta_{t_{n_p}} \right]^T \quad (21)$$

where  $(x_i, y_i, \theta_i)$  is the robot  $i$  pose,  $(x_{c_i}, y_{c_i}, \theta_{c_i})$  is the cluster pose for cluster  $i$ , and  $(x_{t_i}, y_{t_i}, \theta_{t_i})$  is the task-level pose for robot  $i$ . For the position control task, the cluster positions and the task positions are both equated to the robot positions. Accordingly, the forward and inverse kinematic relationships are unity, and the Jacobians are the unit matrix.

For our experiments, the cluster space velocity control law utilizes proportional feedforward and feedback, shown below, for response time and error rejection respectively:

$$\vec{u}_c = K_{c_f} \dot{\vec{c}}_d + K_{c_p} \left( \dot{\vec{c}}_d - \dot{\vec{c}} \right) \quad (22)$$

where  $\vec{u}_c$  denotes cluster space control effort,  $\dot{\vec{c}}_d$  denotes the desired cluster velocity,  $K_{c_f}$  denotes a proportional feedforward gain matrix, and  $K_{c_p}$  denotes a proportional feedback gain matrix. Similarly, the task-space state controller utilizes proportional feedback for error rejection:

$$\vec{u}_t = K_{t_p} \left( \vec{t}_d - \vec{t} \right) \quad (23)$$

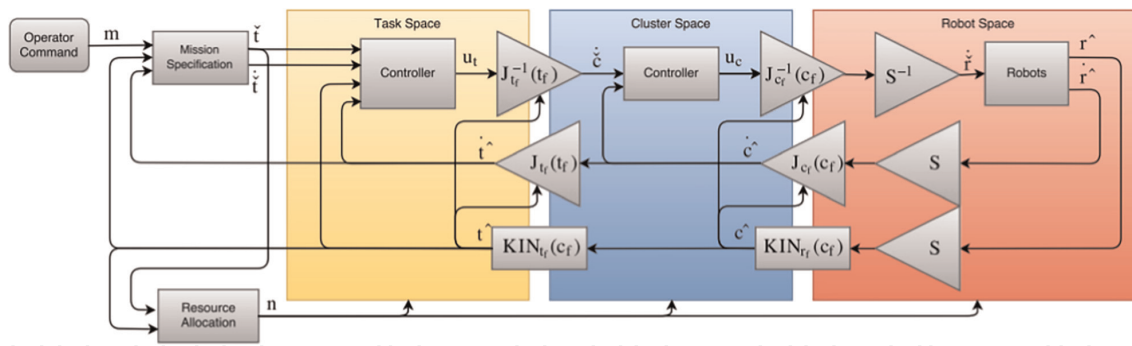
where  $\vec{u}_t$  denotes task space control effort,  $\vec{t}_d$  denotes the desired task state and  $K_{t_p}$  denotes a proportional feedback control gain.

#### 4. Multi-tasking missions

In the previous section, an approach for developing a layered task control architecture for the motion requirements of a multi-robot task is presented. Our interest, however, is not only in task-oriented control of a robot group but also in conducting activities that require multiple interacting tasks, each potentially implemented by a multi-robot group. We term the conduct of such an activity a “mission.” Furthermore, we are interested in collaborative multi-task missions, with “collaboration” implying the ability for one task to support another and for tasks to share resources, such as robots, in an appropriate manner.

To achieve this vision, **Figure 4** shows a control architecture that integrates the two tasks required for the communication relay mission. The architecture integrates the operation of the tasks in two ways. First, on the “front end” of the architecture, a mission-level specification interface provides a mechanism for defining mission-oriented objectives and assigning them to the tasks. In addition, the tasks are allowed to interact, providing a mechanism for tasks to issue commands to and to set constraints for each other. For the communication relay mission, the communication task acts as a master task in order to determine the number of robots it requires; once this determination is made, it specifies the number of remaining robots (to assign to a position hold task) to the position task for position control. In ongoing work with more complex missions, the robot allocation process is independent of any one task. Second, on the “back end” of the architecture, unified motion control is enabled by consolidating the control elements of multiple tasks as described in the following subsection.

Multi-agent systems have the capacity for functional diversity which motivates the ability to change roles during operation. As such, an additional focus of this work investigates the process of reallocating autonomous agents within the given framework. Doing so requires (A) a framework for supporting and transitioning multiple clusters within the same architecture, (B) defining allocation policies to determine when and how to reallocate resources between tasks, and (C) analytic methods to safely transition between robot configurations.



**Figure 4.**  
 A multi-task collaborative Mission control architecture. The diagram shows the consolidated control architecture with additional functions to provide mission specification and resource allocation.

## 4.1 Representation framework

Simultaneously accomplishing multiple mobility tasks requires assigning different tasks to different robots, which, in our framework, implies the use of multiple task-specific robot clusters. While multiple instances of the control framework of Section III could be run in parallel, this approach is static and unable to conveniently support the reconfiguration of clusters as robots are reallocated. Instead, the single-task multi-layered framework is extended to become a multi-task controller that operates on a “federated” system state vector. At a given moment in time, given an assignment of a specific number of robots to specific tasks as defined by the robot allocation vector in (39), the federated kinematic equations and federated Jacobian matrices are composed of the cluster-specific kinematic equations and Jacobian matrices, as shown in (41) and (42).

Robot allocation vector:

$$\vec{n} = [n_1, n_2, \dots, n_o]^T \text{ where } n = \sum_{i=1}^o n_i \quad (24)$$

Federated pose vectors:

$$\vec{r}_f \triangleq [\vec{r}_1, \vec{r}_2, \dots, \vec{r}_o]^T \quad (25)$$

$$\vec{c}_f \triangleq [\vec{c}_1, \vec{c}_2, \dots, \vec{c}_o]^T \quad (26)$$

$$\vec{t}_f \triangleq [\vec{t}_1, \vec{t}_2, \dots, \vec{t}_o]^T \quad (27)$$

Federated kinematic equations:

$$KIN_{c_f}(\vec{c}_f) \triangleq [KIN_{c_1}(\vec{r}_1), KIN_{c_2}(\vec{r}_2), \dots, KIN_{c_o}(\vec{r}_o)]^T \quad (28)$$

$$KIN_{t_f}(\vec{t}_f) \triangleq [KIN_{t_1}(\vec{c}_1), KIN_{t_2}(\vec{c}_2), \dots, KIN_{t_o}(\vec{c}_o)]^T \quad (29)$$

Federated Jacobian matrices:

$$J_{cf} \triangleq \begin{bmatrix} J_{c_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & J_{c_o} \end{bmatrix} \quad (30)$$

$$J_{cf} \triangleq \begin{bmatrix} J_{c_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & J_{c_o} \end{bmatrix} \quad (31)$$

where subscript  $f$  denotes federated elements;  $n_i$  is the number of robots assigned to task  $i$ ;  $\vec{r}_i, \vec{c}_i$  and  $\vec{t}_i$  are the robot space, cluster space, and task space pose vectors for task  $i$ ;  $KIN_{c_i}(\vec{r}_i)$  and  $KIN_{t_i}(\vec{c}_i)$  are the cluster space and task space kinematic equations for task  $i$ ;  $J_{c_i}$  and  $J_{t_i}$  are the cluster space and task space Jacobian matrices for task  $i$ ; and  $o$  is the number of tasks spanning the multi-robot system. The federated pose vector is formulated by concatenating pose vectors, and similarly for the federated kinematic equations. The federated Jacobian matrix is block-diagonal, comprised of uncoupled Jacobian matrices. As the agents shift between tasks, individual elements change size, but the size of the federated elements and the overall structure of the mission-level control system remain constant.

As a simple example of multitasking, consider combining the two previously described tasks into the following two-task mission: maintain communication between two end points or otherwise move to an idle parking position. A subset of the federated elements are shown below for two configurations of a  $n = 3$  robot system:

(1) one robot is allocated to the communications task and two robots are idle ( $\vec{n} = [n_{com}, n_{idle}]^T = [1, 2]^T$ ) and (2) two robots are allocated to the communications task and one robot is idle ( $\vec{n} = [n_{com}, n_{idle}]^T = [2, 1]^T$ ):

Federated cluster pose vector:

$$\vec{c}_M = \begin{cases} \left[ [x_c, y_c, \theta_c] [x_{I_1}, y_{I_1}, \theta_{I_1}, x_{I_2}, y_{I_2}, \theta_{I_2}] \right]^T & \text{for } \vec{n} = [1, 2]^T \\ \left[ [x_c, y_c, \theta_c, \rho_1, \alpha_1, \phi_1] [x_{I_1}, y_{I_1}, \theta_{I_1}] \right]^T & \text{for } \vec{n} = [2, 1]^T \end{cases} \quad (32)$$

## 4.2 Allocation policies

Given the general desire to accommodate multiple tasks and the ability to reassign robots between tasks, we need to incorporate an allocation function for assigning specific robotics to specific tasks. The allocation problem is well studied [38, 40, 41], and we are not proposing any particular innovations in this area; rather, our interest is in determining how any such allocation policy fits into the proposed architecture. To date, we believe that the necessary interface consists of providing the allocation policy with the commanded and actual task state vector such that it can compute  $\vec{n}$ , which is consistent with the approaches discussed in [38]. This value is then provided to each layer of the control architecture in order for controllers and kinematic transforms with appropriate internal dimensions to be selected.



To explore this set of interfaces, we have adopted a simple-but-common state-machine based allocation policy for the communication-idle task allocation process. In particular, we define an aggregate QoS metric known as the chain capacity [27]:

$$\delta \triangleq \frac{n_{com} + 1}{\sum_{i=1}^{n_{com}+1} \frac{1}{s_i}} \quad (33)$$

which we use for the transition function:

$$[n_1 \ n_2] = \begin{cases} [n_1 + 1 \ n_2 - 1], & \text{if } \delta < k_1 \\ [n_1 - 1 \ n_2 + 1], & \text{if } \delta > k_2 \\ [n_1 \ n_2], & \text{otherwise} \end{cases} \quad (34)$$

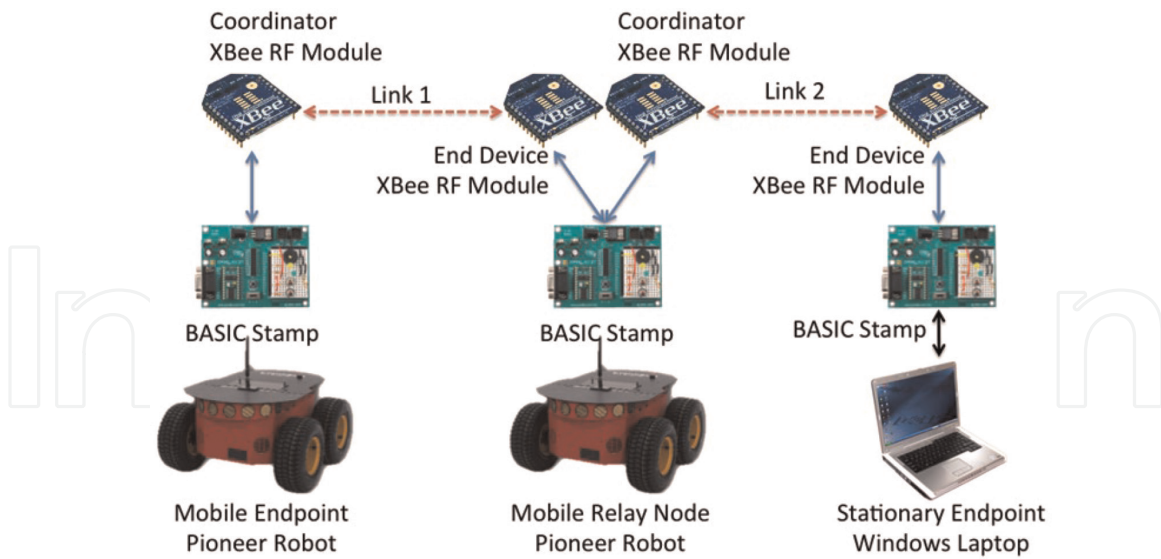
where  $k_1$  and  $k_2$  are performance thresholds that together provide switching hysteresis. For the communication systems and scenarios explored later in this paper,  $k_1$  and  $k_2$  were set to values equivalent to the line-of-sight link power at 40 m and 10 m, respectively. For example, if  $n = [2 \ 1]$  and the signal strength was measured  $s = [-75 \ -76 \ -78]^T$  dBm giving  $\delta = -76.4$  dBm, and the thresholds were  $k = [-52 \ -76]^T$  dBm, then the next robot allocation would be  $n = [3 \ 0]$ .

To reiterate, this policy sets the number of robots to achieve a given link quality while the task-level control provides link balance as stated in (11). This policy is appropriate for the communication relay application and it is also consistent with the general interface requirements hypothesized in the previous paragraph. Future researchers can incorporate the allocation policy that is most appropriate for their tasks, whether state of the art techniques (see [38]) or commonplace yet effective techniques like state machines.

## 5. Experimental testbed

Experimental work used the proven [35] SCU multi-robot test bed, with a communication relay payload. This student-developed system consists of several Pioneer 3-AT skid steered robots with a custom suite of avionics and a centralized off-board control workstation. Wireless 28.8 kbps Ricochet modems are used to relay robot drive commands and position data between the control workstation and the robots. BasicX microcontrollers route drive commands to the robot's built-in speed controller and collect data from a Garmin GPS18 unit and a Devantech CMPS10 compass. Based on experimental evaluation, robot velocity dynamics are approximated as a second-order system with  $\zeta = 0.7$  and  $\omega_n = 2\pi 0.25$  rad/s; simulations used these speed response characteristics for each robot.

Within the control workstation, an open source real-time data streaming server, known as the DataTurbine, relays information between MATLAB/Simulink and simple applications that handle serial port data flow to/from the wireless modems. Controllers execute in real-time within Simulink; this promotes rapid, iterative development in the field and supports rudimentary operator interfaces. Using a dual-core laptop computer, the system maintains a 5 Hz servo loop rate.



**Figure 5.**  
Components in the multi-robot communications relay test bed.

Each robot carries a communications relay payload comprised of two Digi International Xbee Series 2 wireless transceivers connected by a BASIC Stamp microcontroller, shown in **Figure 5**. The microcontroller appends the received signal strength indicator (RSSI) value to each message prior to retransmission. When the end station ultimately receives the message, it also obtains the RSSI state for the multi-hop link. This state data is provided to the task layer controller in order to determine how to adjust the position of the relay cluster. Messages and the associated RSSI measurements were generally executed at 1 Hz, but dropouts due to the inexpensive hardware often resulted in short periods of slower execution, adding a realistic challenge to the experiment.

## 6. Experimental results

A number of simulations and experiments were executed to demonstrate functionality of the system and to showcase particular advantages of the control architecture. First, two scenarios were examined with the single communications task: (A) an experiment showing the single robot system response to hardware configuration changes such as reductions in transmission power, and (B) a simulation showing multi-robot system response to a mobile end station and environmental attenuation. Next, three scenarios were examined performing multiple tasks of communications and idle position control, thereby allowing robots to be added or removed from the communication task: (C) an experiment showing responses to desired link quality commands that require robot reallocation, (D) a simulation showing system response to a moving mobile end station in which robot reallocation is required, and (E) an experiment showing the same capability as in D. Experimental work has demonstrated that the architecture is tolerant of real-world phenomena such as sensor noise, quantization, model mismatches, and communication delays. Simulations allowed rapid exploration with higher numbers of robots, allowing clear demonstration of the architecture behavior without hardware constraints.

## 6.1 Experiment: single task single robot behavior with hardware configuration change

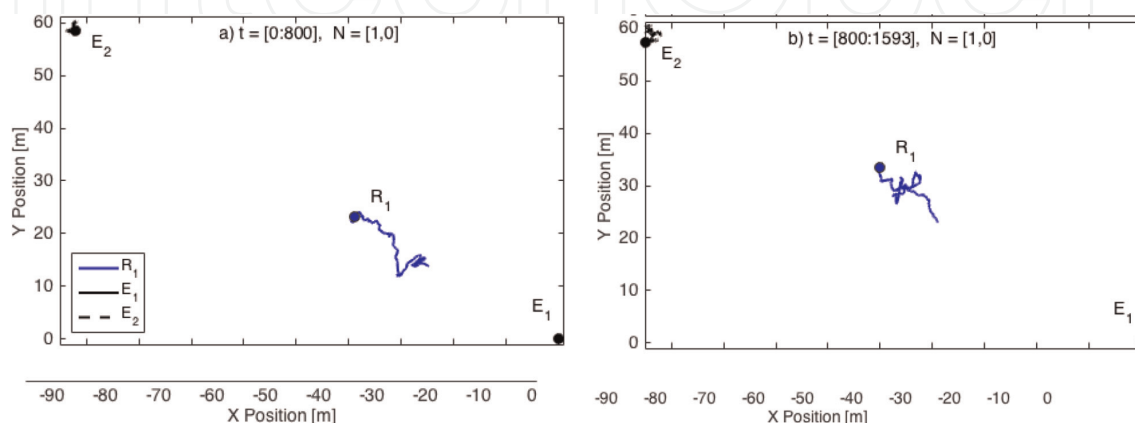
This scenario examines the control system response to configuration changes such as component degradation or a power reduction used to conserve energy. One robot is used to relay communications between two fixed end stations. The system is commanded to achieve unity link balance with null crosstrack error. In this experiment, the system initially moves to an equilibrium position given a nominal communications configuration. Then, at  $t = 800$  sec, the transmission power of end station  $E_2$  is reduced, and the relay robot moves to achieve link equilibrium.

An overhead view of robot position is shown in **Figure 6** where each subplot corresponds to a different time window; in the first, the robot moves to an equilibrium position, and in the second, the robot adjusts its position to balance the link given the power reduction at station  $E_2$ . In **Figure 7** the RSSI values and the link balance parameter (commanded to 1) are shown. In each phase, the systems moves to achieve the commanded link balance, resulting in balanced RSSI values.

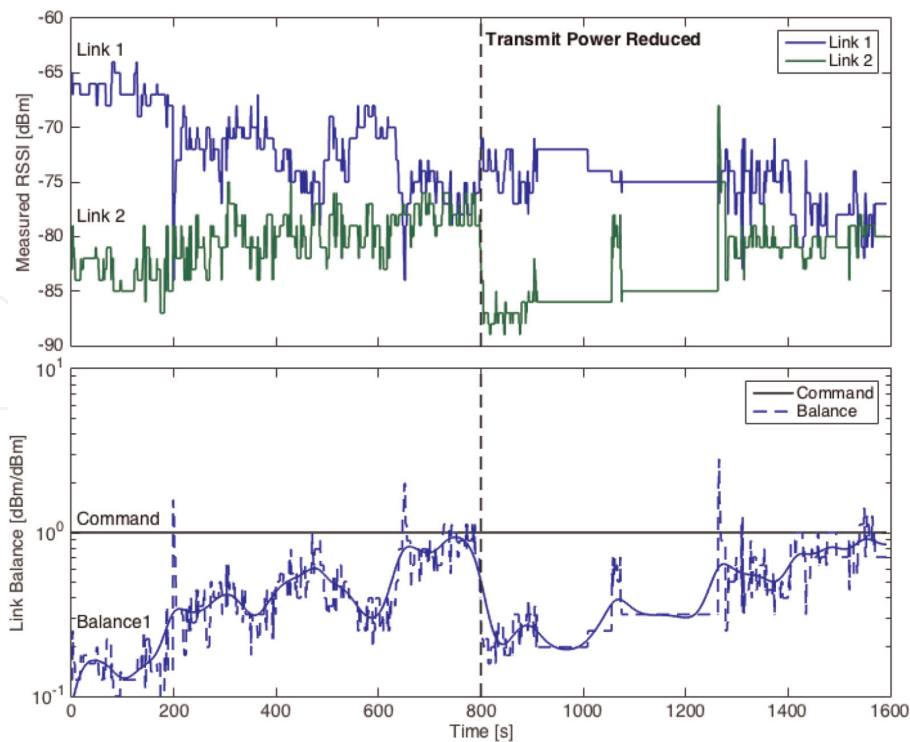
## 6.2 Simulation: single task multi-robot behavior with a mobile end station and local attenuation

This scenario evaluates system behavior given local attenuation effects such as obstructions, fog, or foliage. Three robots are used to relay communications between a fixed base station and a mobile end-node. A comparison is made of the system's performance with and without measurement compensation for these effects to demonstrate how our communications task improves performance compared to the use of a simple link model.

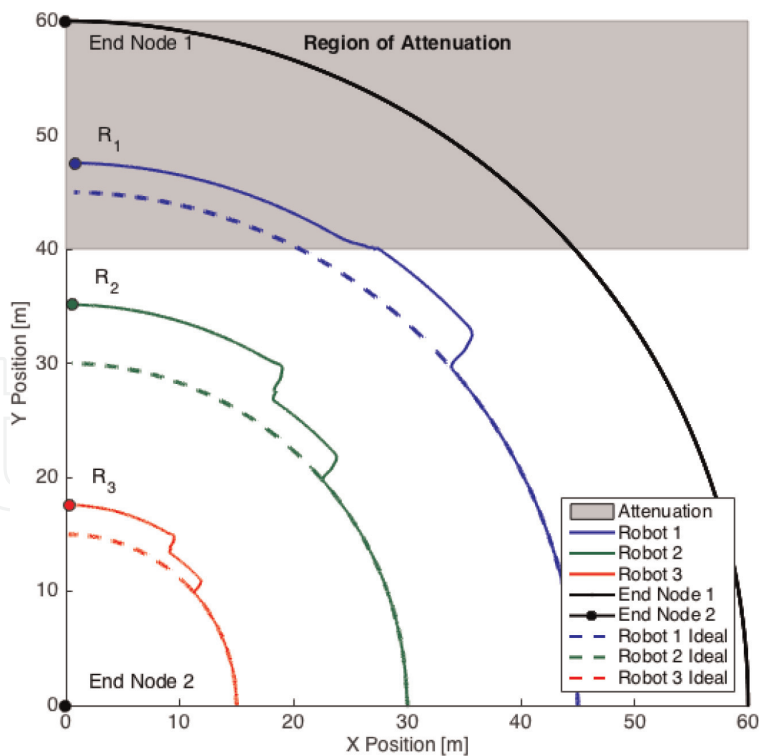
A overhead view is shown in **Figure 8** with robot trajectories plotted for both ideal and attenuated scenarios. Messages are relayed between the base node, located at the origin of the plot, and the mobile end-node, which has a quarter-circle trajectory plotted in black and running from (60,0) to (0,60). A region of power attenuation exists for  $y > 40$ , where any link involving a robot within this area is reduced by half. As the remote end-node traverses its arc at a constant speed, a three-robot cluster maintains link balance as described before. In the ideal case, the robots spread evenly and follow the traverse in concentric arcs, consistent with a model-based approach. In



**Figure 6.**  
 Overhead view of positions of robot  $R_1$  and fixed end nodes  $E_1$  and  $E_2$  at specified times during hardware configuration change experiment.



**Figure 7.**  
Time history of link power and balance ratio during hardware configuration change experiment.



**Figure 8.**  
Overhead view of robot  $R$  positions comparing trajectories in ideal transmission environments (dashed) and trajectories responding to an encountered region of attenuation (solid).

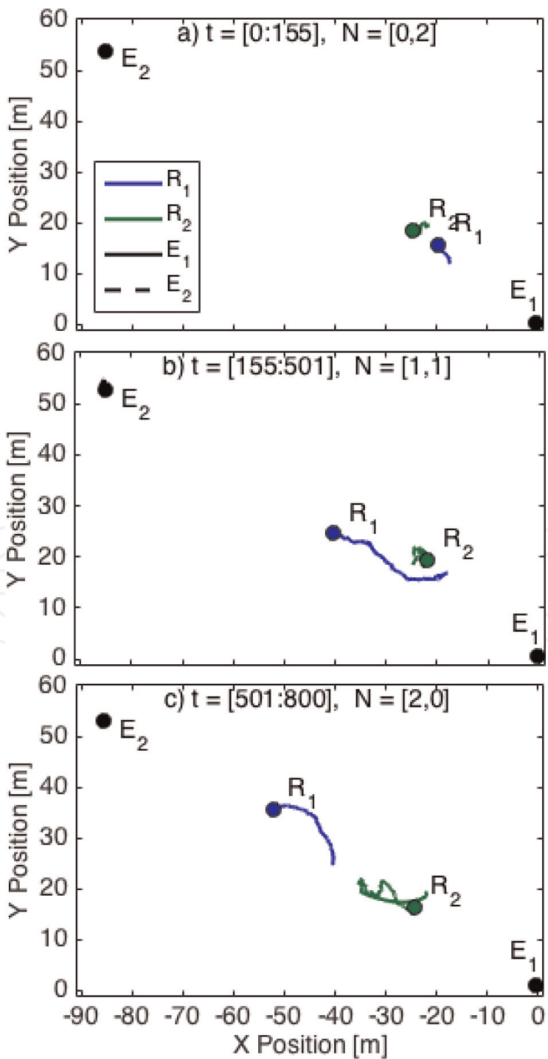
the non-ideal case with attenuation, the multi robot system begins as before, but alters its trajectory to rebalance the links when it senses a drop in signal strength as nodes enters the region of attenuation.

This example demonstrates the value of direct measurement of communication states and high-level task-space control. Sensing the signal strength allows the system to maintain the desired state despite unanticipated characteristics of the environment. In contrast, an open-loop, model-based approach would evenly distribute the nodes as shown, yielding lower performance in non-ideal environments.

### 6.3 Experiment: multi-task multi-robot behavior with configuration link quality command response

This scenario demonstrates changing user requirements for better connectivity or higher throughput thereby forcing a change in the cluster configuration. With fixed communication endpoints and an increase in the commanded link quality, two robots are sequentially reallocated from the idle task to the communication relay task. Each newly incorporated robot moves from its idle position to a location determined through execution of the communication task in order to achieve the commanded link quality and link balance set-points while the other robots in the communication task reposition themselves accordingly.

**Figure 9** shows the fixed end nodes  $E_1$  and  $E_2$  and two mobile robots. In the top plot, for time  $t = [0 : 155]$ , the commanded link quality is such that the two end



**Figure 9.**  
Overhead view of positions of robots  $R$  in response to a link quality command between end nodes  $E$ .



stations communicate directly with each other without the need for a relay node. As a result, the configuration is  $\vec{n} = [0, 2]^T$ , with both robots assigned to an idle task and holding their position. As seen in **Figure 10**, link quality is maintained within an acceptable deadband; the link balance is not shown given the single hop.

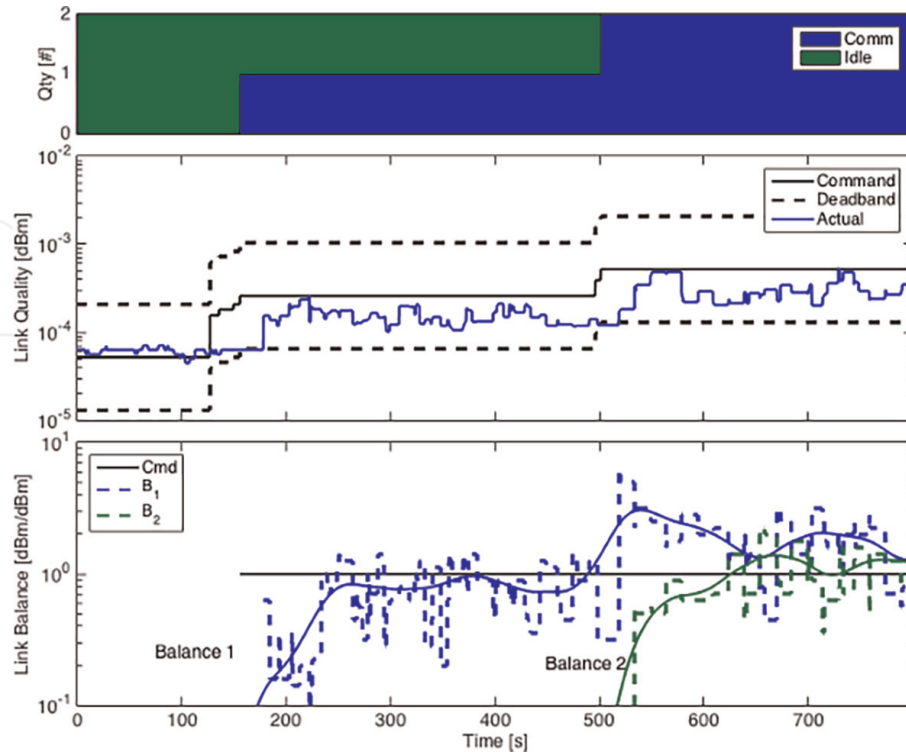
For time  $t = [155 : 501]$ , the link quality setpoint is increased. As seen in **Figure 10b**, the link quality deadband is violated, leading to a change in the assignment of tasks such that  $\vec{n} = [1, 1]^T$ , shown in **Figure 10a**. This results in controlling  $B_1$ , the link balance between the two existing sub-links, by achieving balance as shown in **Figure 10c** by the repositioning of robot 1, as shown in **Figure 9b**.

During time  $t = [501 : 800]$ , the system responds to another command to increase link quality. Again, the link quality deadband is violated, leading to the addition of robot  $R_2$  to the communication task. With two intermediate robots, two link balancing operations take place in parallel, shown in **Figure 10c**. The initial position of the added robot creates a large switching transient in the link balance. Accordingly, the robots alter their positions, as shown in **Figure 9c**.

The plots in **Figure 10** show that the sensed RSSI parameters were clearly not ideal, exhibiting noise and quantization and the effects of a wide range on non-ideal characteristics of the wireless links. Because of these non-ideal characteristics, the robots do not move to the geometric center of the end points. These real-world phenomena are challenging but the control architecture is sufficiently robust to tolerate these unmodeled effects.

#### 6.4 Simulation: multi-task multi-robot behavior with a mobile end station

This simulation demonstrates control of link quality and balance with a mobile endpoint, gracefully adding and removing robots as appropriate for the task. The

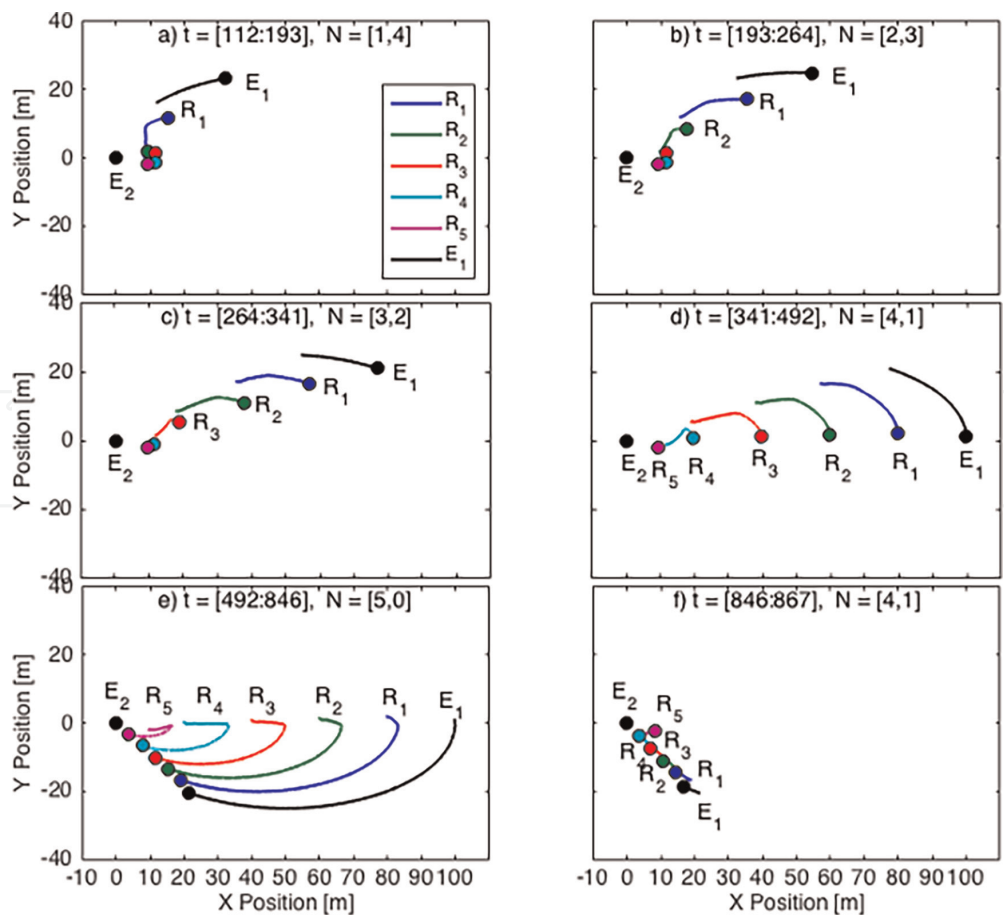


**Figure 10.** Time history of key states while evaluating the link quality commanded response, forcing configuration change.

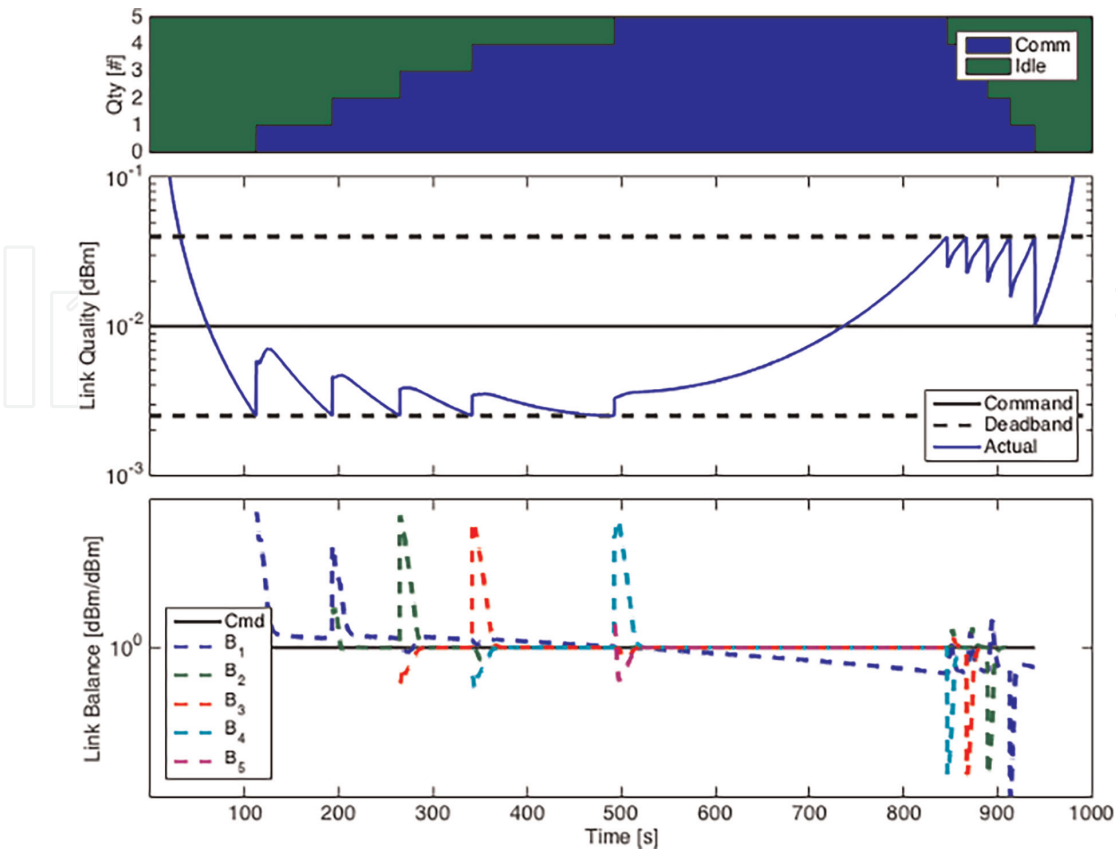
scenario starts with a mobile end station communicating directly with a fixed end station and with five robots executing the idle position hold task. As the mobile end station executes an elliptical trajectory that moves it away from and then closer to the fixed end station, the five relay robots are sequentially added and then removed to the communication task, maintaining the specified link quality and balance (**Figure 11**).

The evolution of the system's state can be seen in **Figure 12**. For time  $t = [0,500]$ , the end station moves away from the fixed end station, lowering link quality, as shown in **Figure 12b**. At times  $t = \sim 110$  sec, 193 sec, 265 sec,  $\sim 340$  sec and  $\sim 490$  sec, the link quality hits the threshold for acceptable link quality. At each of these times, the controller re-assigns a new robot from the idle task to the communications task, thereby causing  $\vec{n}$  to change at these times, as seen in **Figure 12a**, and new robots being added to the multi-hop link, as seen in **Figure 12c**; idle robots are controlled to remain in their default position. These additions to sub-links in the communication chain lead to new balance parameters to be controlled,  $B_1$  through  $B_5$ .

As the mobile end station turns back towards the base node, the link quality increases. Each time this value hits the high deadband, at times  $t = 845$  sec,  $\sim 870$  sec,  $\sim 890$  sec,  $\sim 910$  sec, and  $\sim 935$  sec, an active communication task robot is returned to the idle task. This reduces the number of link balance parameters, leading to a transient in link balance that is quickly controlled through the repositioning of the remaining communication task robots. Interestingly, the deadband causes unequal times between transitions as the robots are faster to move out due to the task state definition and allocation policy and slower to move into the communication cluster.



**Figure 11.**  
Overhead view of robots  $R$  and end nodes  $E$  during specified times for mobile endpoint simulation.



**Figure 12.**  
Time history of key system states for mobile endpoint simulation.

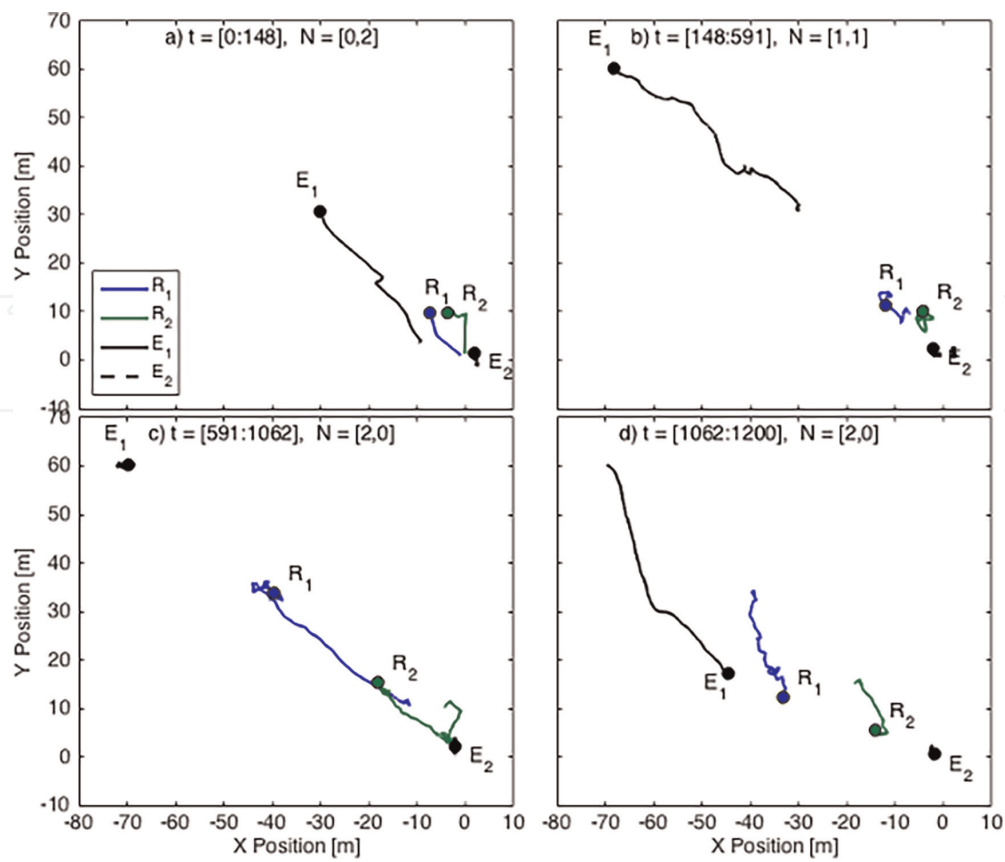
This demonstrates the ability of the control architecture to respond to motion of the end node based on sensed link characteristics and to reallocate robots without any additional commanding.

## 6.5 Experiment: Multi-task multi-robot behavior with a mobile end station

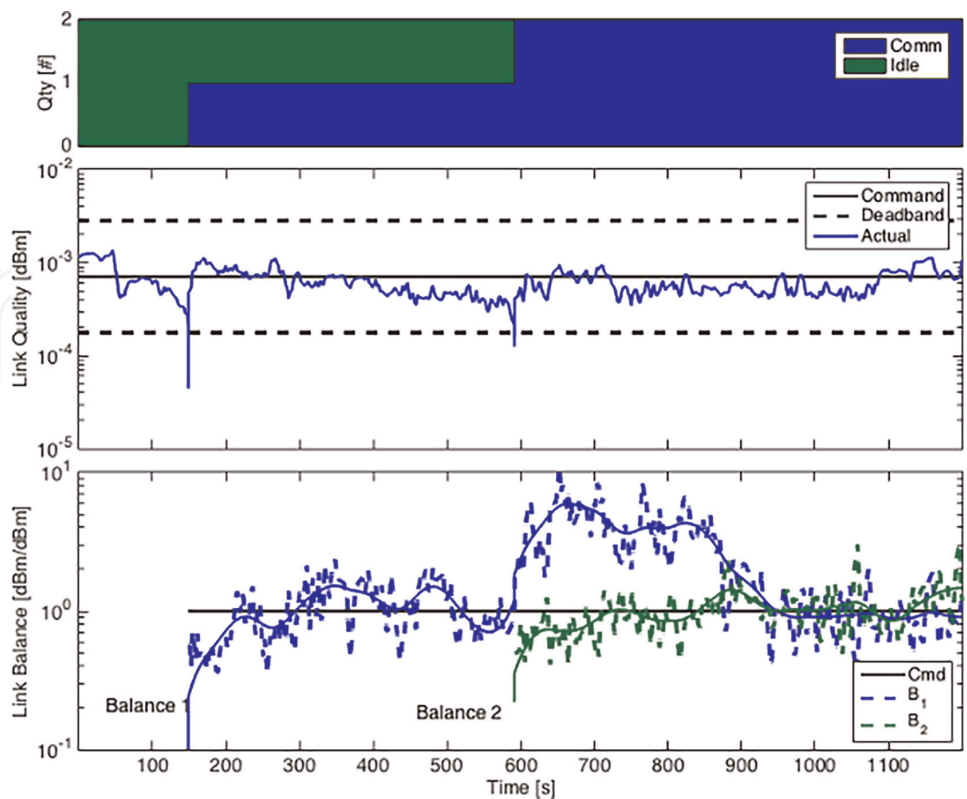
Like the simulation presented in Section VI.D, this experiment demonstrates the control of link quality and balance with a mobile end node. The experiment starts with the end stations near each other and directly communicating, with two relay robots in an idle position. As the mobile end station moves away, relay robots are sequentially added to maintain the specified level of link quality and balance.

**Figure 13** shows the paths taken by the robots and endpoints, and **Figure 14** shows the corresponding state trajectories. In **Figure 13a**, for time  $t = [0 : 148]$ , the mobile endpoint can be seen moving away from the stationary endpoint while the link quality remains within the deadband. The communication relay robots are allocated to idle,  $\vec{n} = [0, 2]^T$ , and can be seen parking themselves.

At time  $t = 148$ , the link quality exceeds the lower bounds of the deadband and the allocation policy adds a robot to the communication relay task, changing the configuration vector to  $\vec{n} = [1, 1]^T$ . In **Figure 13b**, for time  $t = [148 : 591]$ , the new robot relay moves to balance the communication links while the mobile end station continues moving away from the stationary endpoint. Though there is not significant movement of the relay robot, the measured link states, shown in **Figure 14**, indicate that the balance setpoint is achieved during this time. This demonstrates the complexity and non-intuitiveness of RF fields and the benefit of communication-space measurement



**Figure 13.**  
Overhead view of robots  $R$  and end nodes  $E$  during specified times for mobile endpoint experiment.



**Figure 14.**  
Time history of key system states for mobile endpoint experiment.

and control, including the effects of the deadbands; alternatively locating the relay robots in the geometric center of the two points would yield worse performance.

At time  $t = 591$ , the link quality again exceeds the lower bounds of the deadband and the allocation policy adds the second robot to the communication relay task, changing the configuration vector to  $\vec{n} = [2, 0]^T$ . In **Figure 13c**, for time  $t = [591 : 1062]$ , both robots move to balance the communication links. The switching transient can be seen in **Figure 14** starting at  $t = \sim 600$  sec and settling by  $t = \sim 950$  sec. The final overhead plot, **Figure 13d**, shows the mobile endpoint arcing back towards the stationary endpoint and the relay robots mimic its motion to maintain link balance.

## 7. Future work and summary

In this article, we presented a multi-robot control architecture providing explicit task control for improved performance yet with abstraction from implementation. Direct sensing and operational task space control eliminate errors due to modeling and implicit specification. In addition, controllers can compensate for non-ideal behavior in the appropriate space of the layered architecture. Abstraction provides the flexibility to engage different types and quantities of robots to accomplish tasks. This segregates individual task complexity in order to facilitate large-scale missions with many tasks. We proposed a design methodology for composing new spatially-sensitive tasks that includes conditions for stability and quantification of responsiveness. The architecture was demonstrated using the example task of communication. Experiments and simulations exhibited explicit control of task states, compensating for the complex behavior inherent in real-world communication networks. The system successfully reacts to a dynamic environment, varying operator commands, and hardware configuration changes.


Ongoing work leverages this architecture for larger missions comprised of more tasks with more complex interactions. This article is a step towards multi-task missions indicative of systems of systems. Continued development of this architecture with new applications and new environments increases the utility of integrated multi-agent systems.

### Author details

John Shepard and Christopher Kitts\*  
Santa Clara University, Santa Clara, CA, USA

\*Address all correspondence to: ckitts@scu.edu

### IntechOpen

© 2023 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 



## References

- [1] Joordens MA, Jamshidi M. Consensus control for a system of underwater swarm robots. *IEEE Systems Journal*. 2022;**4**(1):65-73
- [2] Balch T, Arkin RC. Behavior-based formation control for multi-robot teams. *IEEE Transactions on Robotics and Automation*. 1998;**14**(6):926-939
- [3] Ailon A, Zohar I. Control strategies for driving a group of nonholonomic kinematic mobile robots in formation along a time-parameterized path. *IEEE/ASME Transactions on Mechatronics*. 2012;**17**(2):326-336
- [4] Mahacek P, Kitts C, Mas I. Dynamic guarding of marine assets through cluster control of automated surface vessel fleets. *IEEE/ASME Transactions on Mechatronics*. 2012;**17**(1):65-75
- [5] Mas I, Li S, Acain J, Kitts CA. Entrapment/escorting and patrolling missions in multi-robot cluster space control. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*; St. Louis, USA. New York City, NY, USA: IEEE; 2009
- [6] Elston J, Frew EW. Hierarchical distributed control for search and tracking by heterogeneous aerial robot networks. In: *IEEE International Conference on Robotics and Automation*; Pasadena, CA, USA. New York City, NY, USA: IEEE; 2008
- [7] Adamek T, Kitts C, Mas I. Gradient-based cluster space navigation for autonomous surface vessels. *IEEE/ASME Transactions on Mechatronics*. 2014;**20**(2):506-518
- [8] Li S, Kong R, Guo Y. Cooperative distributed source seeking by multiple robots: Algorithms and experiments. *IEEE/ASME Transactions on Mechatronics*. 2014;**19**(6):1810-1820
- [9] Jevtic A, Gutierrez A, Andina D, Jamshidi M. Distributed bees algorithm for task allocation in swarm of robots. *IEEE Systems Journal*. 2012;**6**(2):296-304
- [10] Ross A, Layton T, Romanishin J, Rus D. IkeaBot: An autonomous multi-robot coordinated furniture assembly system. In: *IEEE International Conference on Robotics and Automation*; Karlsruhe, Germany. New York City, NY, USA: IEEE; 2013
- [11] Yoon D-K, Seo J-T, Yi B-J. Automatic lighting system using multiple robotic lamps. *IEEE/ASME Transactions on Mechatronics*. 2014;**19**(3):963-974
- [12] Okamoto G, Chen C, Kitts C. Beamforming performance for a reconfigurable sparse array smart antenna system via multiple mobile robotic systems. In: *Proc of the SPIE - The International Society for Optical Engineers*. 2010
- [13] Tortora G, Dario P, Menciassi A. Array of robots augmenting the kinematics of endocavitary surgery. *IEEE/ASME Transactions on Mechatronics*. 2014;**19**(3):1821
- [14] Zemiti N, Morel G, Ortmaier T, Bonnet N. Mechatronic design of a new robot for force control in minimally invasive surgery. *IEEE/ASME Transactions on Mechatronics*. 2007;**12**(2):143-153
- [15] Ansola PG, Higuera AG, Otamendi FJ, Morenas J. Agent-based distributed control for improving complex resource scheduling: Application to airport

ground handling operations. *IEEE Systems Journal*. 2014;**8**(4):1145-1157

[16] Grocholsky B, Keller J, Kumar V, Pappas G. Cooperative air and ground surveillance. *IEEE Robotics and Automation Magazine*. 2006;**2006**:16-26

[17] Kudelski M, Gambardella LM, Di Caro GA. A mobility-controlled link quality learning protocol for multi-robot coordination tasks. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). 2014. pp. 5024-5031

[18] Kim J, Ladosz P, Oh H. Optimal communication relay positioning in mobile multi-node networks. *Robotics and Autonomous Systems*. 2020;**129**: 103517

[19] Pezeshkian N, Nguyen HG, Burmeister A. Unmanned ground vehicle radio relay deployment system for nonline-of-sight operations. In: Proceedings of the 13th IASTED International Conference on Robotics and Applications (RA '07). San Diego CA, USA: Space and Naval Warfare Systems Center; 2007. pp. 501-506

[20] Ollero A, Marron PJ, Bernard M, et al. AWARE: Platform for autonomous self-deploying and operation of wireless sensor-actuator networks cooperating with unmanned AeRial vehicleS. In: Proc of IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR '07). New York City, NY, USA: IEEE; 2007. pp. 1-6

[21] Pezeshkian N, Neff J, Hart A. Link quality estimator for a mobile robot. In: Int. Conf. on Informatics in Control, Automation and Robotics; Rome, Italy. 2012

[22] Mehta VK, Arrichiello F. Ithica, NY, USA: Cornell University; 2013. [Online]. <http://arxiv.org/abs/1312.2526v1>

[23] Sweeney J, Grupen R, Shenoy P. Active QoS flow maintenance in controlled mobile networks. In: The Fourth International Symposium on Robotics and Automation (ISRA); Queretaro, Mexico. 2005

[24] Shepard J, Kitts C. A multi-robot control architecture for collaborative missions comprised of tightly- coupled, interconnected tasks. *IEEE Systems Journal*. 2016;**12**(2):1435-1446

[25] Gao Y, Chen H, Li Y, Lyu C, Liu Y. Autonomous Wi-Fi relay placement with mobile robots. *IEEE/ASME Transactions on Mechatronics*. 2017;**22**(6):2532-2542

[26] Im H-J, Lee C-E, Cho Y-J, Sunghoon K. RSSI-based control of mobile cooperative robots for seamless networking. In: Control, Automation, and Systems, International Conference on; Jeju Island, Korea. New York City, NY, USA: IEEE; 2012

[27] Dixon C, Frew EW. Optimizing cascaded chains of unmanned aircraft acting as communication relays. *IEEE Journal on Selected Areas in Communications*. New York City, NY, USA: IEEE; 2012;**30**(5):883-898

[28] Bryan J. Thibodeau, Andrew H. Fagg, and Brian N. Levine, "Signal Strength Coordination for Cooperative Mapping; Amherst, USA, 2005.

[29] Min BC, Kim Y, Lee S, Jung J, Matson ET. Finding the optimal location and allocation of relay robots for building a rapid end-to-end wireless communication. *Ad Hoc Networks*. Amsterdam, Netherlands: Elsevier; 2016; **39**:23-44

[30] Stephan J, Fink J, Kumar V, Ribeiro A. Concurrent control of mobility and communication in multirobot systems.

IEEE Transactions on Robotics. 2017;**33**  
 (5):1248-1254

[31] Bezzo N, Yan Y, Fierro R, Mostofi Y. A decentralized connectivity strategy for mobile robot swarms. In: 18th World Congress of the International Federation of Automatic Control; Milan, Italy. New York City, NY, USA: IFAC; 2011. pp. 4501-4506

[32] Bezzo N et al. A cooperative heterogeneous mobile wireless mechatronic system. IEEE/ASME Transactions on Mechatronics. New York City, NY, USA: IEEE; 2014;**19**(1): 20-31

[33] Kitts C, Mas I. Cluster space specification and control of mobile multirobot systems. IEEE/ASME Transactions on Mechatronics. 2009;**14** (2):207-218

[34] John J. Craig, Introduction to Robotics: Mechanics and Control. 3rd ed. Pearson Prentice Hall; 2005

[35] Mas I, Petrovic O, Kitts C. Cluster space specification and control of a 3-robot mobile system. In: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on; Pasadena, CA, USA. 2008. pp. 3763-3768

[36] Mas I, Kitts C. Obstacle avoidance policies for cluster space control of nonholonomic multirobot systems. IEEE/ASME Transactions on Mechatronics. New York City, NY, USA: IEEE; 2012;**17**(6):1068-1079

[37] Khatib O. A unified approach for motion and force control of robot manipulators: The operational space formulation. Robotics and Automation, IEEE Journal of. 1987;**3**(1):1068

[38] Kopeikin A, Ponda SS, How JP. Control of communication networks for

teams of UAVs. In: Valavanis KP, Vachtsevanos GJ, editors. Handbook of Unmanned Aerial Vehicles. Springer Science+Business; 2015

[39] Agnew MS, Canto PD, Kitts CA, Li SQ. Cluster space control of aerial robots. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics; Montreal, Canada. New York City, NY, USA: IEEE; 2010

[40] Ayorkor Korsah G, Stentz A, Bernardine Dias M. A comprehensive taxonomy for multi-robot task allocation. The International Journal of Robotics Research. 2013;**32**(12): 1495-1512

[41] Ponda SS, Johnson LB, Kopeikin AN, How JP. Distributed planning strategies to ensure network connectivity for dynamic heterogeneous teams. IEEE Journal on Selected Areas in Communications. June 2012;**30**(5): 861-869