Adaptive Teams of Autonomous Aerial and Ground Robots for Situational Awareness

**Mong-ying A. Hsieh, Anthony Cowley, James F. Keller,
Luiz Chaimowicz, Ben Grocholsky, Vijay Kumar, and Camillo J. Taylor**
GRASP Laboratory
University of Pennsylvania
Philadelphia, PA 19104
{acowley,mya,jfkeller,kumar,cjtaylor}@grasp.cis.upenn.edu

Yoichiro Endo and Ronald C. Arkin
Georgia Tech Mobile Robot Lab
College of Computing
Georgia Institute of Technology
Atlanta, GA 303332
{endo,arkin}@cc.gatech.edu

Boyoon Jung, Denis F. Wolf, and Gaurav S. Sukhatme
Robotic Embedded Systems Laboratory
Center for Robotics and Embedded Systems
University of Southern California
Los Angeles, CA 90089
{boyoon,denis,gaurav@robotics.usc.edu}

Douglas C. MacKenzie
Mobile Intelligence Corporation
Livonia, MI 48150 doug@mobile-intelligence.com

Abstract

In this paper, we report on the integration challenges of the various component technologies developed towards the establishment of a framework for deploying an adaptive system of heterogeneous robots for urban surveillance. In our integrated experiment and demonstration, aerial robots generate maps that are used to design navigation controllers and plan missions for the team. A team of ground robots constructs a radio signal strength map that is used as an aid for planning missions. Multiple robots establish a mobile, ad-hoc communication network that is aware of the radio signal strength between nodes and can adapt to changing conditions to maintain connectivity. Finally, the team of aerial and ground robots is able to monitor a small village, and search for and localize human targets by the color of the uniform, while ensuring that the information from the team is available to a remotely located human operator. The key component technologies and contributions include (a) mission specification and planning software; (b) exploration and mapping of radio signal strengths in an urban environment; (c) programming abstractions and composition of controllers for multi-robot deployment; (d) cooperative control strategies for search, identification, and localization of targets; and (e) three-dimensional mapping in an urban setting.

1 Introduction

Urban and unstructured environments provide unique challenges for the deployment of multi-robot teams. In these environments, buildings and large obstacles pose 3-D constraints on visibility, communication network performance is difficult to predict and GPS measurements can be unreliable or even unavailable. The deployment of a network of aerial and ground vehicles working in cooperation can often achieve better performance since these three-dimensional sensing networks may be better poised to obtain higher quality and more complete information and be robust to the challenges posed by these environments. Under these circumstances, it is necessary to keep the network tightly integrated at all times to enable the vehicles to better cooperate and collaborate and achieve greater synergy. Furthermore, one must provide enabling technologies to permit the deployment of these heterogeneous teams of autonomous mobile robots by a few human operators to execute the desired mission. This paper presents our attempts to realize our vision of an autonomous, adaptive robot network capable of executing a wide range of tasks within an urban environment. The work, funded by the Defense Advanced Research Projects Agency's (DARPA) MARS2020 program, was a collaborative effort between the General Robotics, Automation, Sensing & Perception (GRASP) Laboratory at the University of Pennsylvania, the Georgia Tech Mobile Robot Laboratory and the University of Southern California's (USC) Robotic Embedded Systems Laboratory.

Our vision for the project was the development of a framework that would enable a single human operator to deploy a heterogenous team of autonomous air and ground robots to cooperatively execute tasks such as surveillance, reconnaissance, and target search and localization, within an urban environment while providing high-level situational awareness for a remote human operator. Additionally, the framework would enable autonomous robots to synthesize the desirable features and capabilities of both deliberative and reactive control while incorporating a capability for learning. This would also include a software composition methodology that incorporates both pre-composed coding and learning-derived or automated coding software to increase the ability of autonomous robots to function in unpredictable environments. Moreover, the framework would be context driven, and use multi-sensor processing to disambiguate sensor-derived, environmental state information. A team of heterogeneous robots with these capabilities has the potential to empower the individual robotic platforms to efficiently and accurately characterize the environment, and hence potentially exceed the performance of human agents. In short, our goals for the project were to develop and demonstrate an architecture, the algorithms and software tools that:

- are independent of team composition;
- are independent of team size, i.e. number of robots;
- are able to execute of a wide range of tasks;
- allow a single operator to command and control the team;
- allow for interactive interrogation and/or reassignment of any robot by the operator at the task or team level.

We report on the first outdoor deployment of a team of heterogeneous aerial and ground vehicles which brought together three institutions with over 15 different robotic assets to demonstrate communication sensitive behaviors for situational awareness in an urban village at the McKenna Military Operations on Urban Terrain (MOUT) site in Fort Benning, Georgia. The integrated demonstration was the culmination of the MARS2020 project bringing together the various component technologies developed as part of the project. The demonstration featured four distinct types of ground robots each using different types of command and control software, and operating systems at the platform level. These were coordinated at the team level by a common mission plan and operator control and display interconnected through an ad-hoc wireless network. The result was an integrated team of UAVs and UGVs, in which the team and the network had the ability to adapt to the needs and commands of a remotely located human operator to provide situational awareness.

This paper is organized as follows: We present some related work in networked robotic systems in Section 2. Section 3 provides a brief description of the experimental testbed used to evaluate the component technologies summarized in Section 4. Section 5 describes the integrated demonstration that brought together the numerous key technologies summarized in this paper and the integration challenges. Section 6 provides a discussion on the successes and lessons learned with some concluding remarks.

2 Related Work

There have been many successes in the manufacturing industry where existing sensors, actuators, material handling equipment and robots have been reconfigured and networked with new robots and sensors via wireless networks to enhance productivity, quality and safety. However, in most of these cases, the networked robots operate in a structured environment with very little variation in configuration and/or operating conditions and tasks are often well-defined and self-contained.

The growing interest in the convergence of the areas of multi-agent robotics and sensor networks have lead to the development of networks of sensors and robots that not only can perceive their environment but also achieve tasks such as locomotion [Majumder et al., 2001], manipulation [Kang et al., 2000], surveillance [Hsieh et al., 2006], and search and rescue to name a few. Besides being able to perform tasks that individual robots cannot perform, networked robots also result in improved efficiency. Tasks like searching or mapping [Thibodeau et al., 2004] can be achieved by deploying multiple robots performing operations in parallel in a coordinated fashion. Furthermore, networked systems enable fault-tolerance in design by having the ability to react to information sensed by other mobile agents or remote sensors. This results in the potential to provide great synergy by bringing together components with complementary benefits and making the whole greater than the sum of the parts.

Some applications for networked robots include environmental monitoring, where one can exploit mobility and communication abilities of the robotic infrastructure for observation and data-collection at unprecedented scales in various aspects of ecological monitoring. Some examples include [Sukhatme et al., 2006] for aquatic monitoring, [Kaiser et al., 2005] for terrestrial monitoring, and [Amarss, 2006] for subsoil monitoring. Other applications for networked robotic systems include surveillance of indoor environments [Rybski et al., 2000] and support for first responders in a search and rescue operation [Kotay et al., 2005]. In [Corke et al., 2003], the communication capabilities of a network of stationary sensor nodes are exploited to aid in the localization and navigation of an autonomous aerial vehicle, while [Durrant-Whyte et al., 2001] exploits the parallel processing power of sensor networks for data fusion. A theoretical framework for controlling team formation for optimal target tracking is provided in [Spletzer and Taylor, 2002] while [Stroupe and Balch, 2003] uses a behavior-based approach to solve a similar problem. In [Sukkarieh et al., 2003], cooperative target tracking is achieved by optimizing over all joint team actions.

While there are many successful embodiments of networked robots with numerous applications there are significant challenges that have to be overcome. The problem of coordinating multiple autonomous units and making them cooperate creates problems at the intersection of communication, control and perception. Cooperation entails more than one entity working toward a common goal while coordination implies a coupling between entities that is designed to achieve the common goal. Some works that consider coordination and task allocation strategies in uncertain environments include [Mataric et al., 2003], [Lerman et al., 2006], and [McMillen and Veloso, 2006]. A behavior-based software architecture for heterogeneous multi-robot cooperation is proposed in [Parker, 2005], while a methodology for automatic synthesis of coordination strategies for multi-robot teams to execute given tasks is described in [Tang and Parker, 2005]. A market-based task allocation algorithm for multi-robot teams tasked to extinguish a series of fires arising from some disaster is considered in [Jones et al., 2006b]. Dynamic coalition formation for a team of heterogeneous robots executing tightly coupled tasks is considered in [Jones et al., 2006a].

Our goal is to develop networks of sensors and robots that can perceive their environment and respond to it, anticipating information needs of the network users, repositioning and self-organizing themselves to best acquire and deliver the information, thus achieving seamless situational awareness within various types of environments. Furthermore, we are also interested in providing proper interfaces to enable a single human user to deploy networks of unmanned aerial, ground, surface and underwater vehicles. There have been several recent demonstrations of multi-robot systems exploring urban environments [*et. al.*, 2005, Grocholsky et al., 2005] and interiors of buildings [Howard et al., 2006, Fox et al., 2006] to detect and track intruders, and transmit all of the above information to a remote operator. Although these examples show that it is possible to deploy networked robots using an off-the-shelf 802.11b wireless network and have the team be remotely tasked and monitored by a single operator, they do not quite match the level of team heterogeneity and complexity described in this paper.

3 Experimental Testbed

Our multi-robot team consists of two unmanned aerial vehicles (UAVs) and 8 unmanned ground vehicles (UGVs). In this section, we provide a short description of the various components of the experimental testbed used to evaluate the key technologies employed in the integrated experiment.

3.1 UAVs

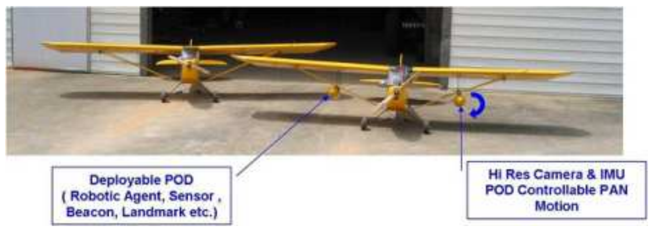
The two UAVs are quarter scale Piper Cub J3 model airplanes with a wing span of 104 inches ($\sim$ 2.7 m) (see Figure 1(a)). The glow fuel engine has a power rating of 3.5 HP, resulting in a maximum cruise speed of 60 knots ($\sim 30m/s$), at altitudes up to 5000 feet ($\sim$ 1500 m), and a flight duration of 15 - 20 minutes. Each UAV is equipped with a sensor pod containing a high resolution firewire camera, inertial sensors and a 10Hz GPS receiver (See Figure 1(b)) and is controlled by a highly integrated, user customizable Piccolo avionics board which is manufactured by *CloudCap Technologies* [Vaglienti and Hoag, 2003]. The autopilot provides innerloop attitude and velocity stabilization control allowing research to focus on guidance at the mission level.

Additionally, each UAV continuously communicates with the ground station at 1 Hz and the range of the communication can reach up to 6 miles. Direct communication between UAVs can be emulated through the ground or using the local communication channel on the UAVs (802.11b - wireless network card). The ground station has an operator interface program (shown in Figure 2), which allows the operator to monitor flight progress, obtain telemetry data, or dynamically change the flight plans using geo-referenced maps. The ground station can concurrently monitor up to 10 UAVs and performs differential GPS corrections and updates the flight plan, which is a sequence of three dimensional way-points connected by straight lines.

3.2 UGVs

Our team of UGVs consist of 3 ClodBusters, 2 Pioneer2 ATs, 1 Segway RMP, 2 ATRV-Jrs, and an AM General Hummer Vehicle modified and augmented with multiple command and control computers and deployed as a Base Station. The ClodBuster UGVs, are commercial 4WD model trucks modified and augmented with a Pentium III laptop computer, specially designed Universal Serial Bus (USB) device which controls drive motors, odometry, steering servos and a camera pan mount with input from the PC, GPS receiver, IMU and firewire stereo camera. The Pioneer2 AT is a typical four-wheeled, statically stable robot designed for outdoor uses. This skid-steer platform can rotate in place and achieve a maximum speed of 0.7 meters per second. The Segway RMP is a two-wheeled, dynamically stable robot with self-balancing capability. Both the Pioneer2 AT and the Segway are equipped with a GPS receiver, an IMU, built-in odometry, a horizontal scanning laser sensor, and a pan/tilt/zoom-

(a)



(b)

Figure 1: (a) Two Piper J3 cub model airplanes. (b) UAV external payloads (POD).

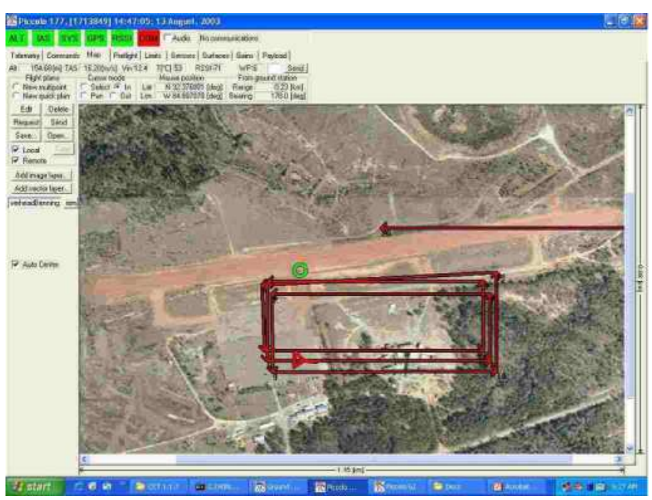


Figure 2: Ground Station Operator Interface showing flight plan and actual UAV position. (August 2003, Fort Benning, Georgia)

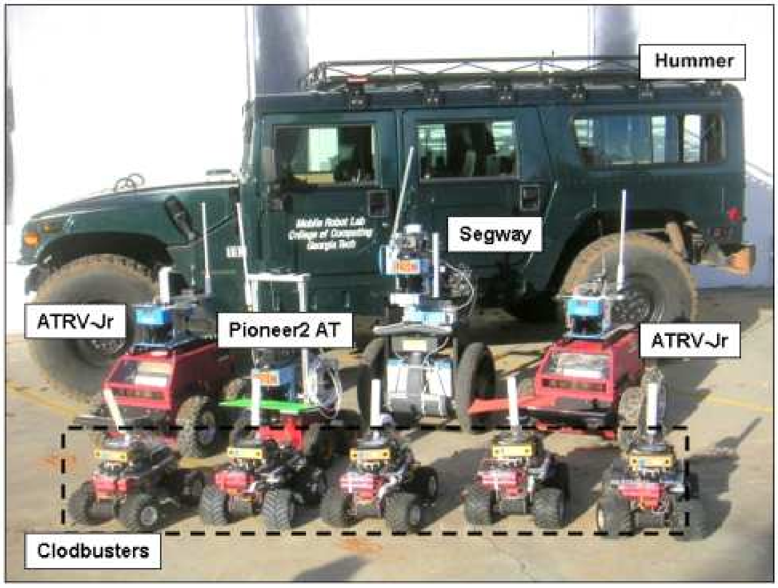


Figure 3: Our team of unmanned ground vehicles.

capable camera. The Segway is also equipped with an additional vertical scanning laser to enable 3-dimensional mapping.

The ATRV-Jr is a four-wheeled robot that can navigate outdoor terrains reaching approximately 2 meters per second at its full speed. It is equipped with onboard dual processor Pentium III computers, a differential GPS , a compass, an IMU, and shaft-encoders. In addition, two sets of laser range finders are mounted on top of the robot in order to provide full 360-degree coverage for obstacle detection. The Hummer Vehicle is outfitted with seating for three human operators and command and control computers for UGV deployment, launch missions, and monitor the progress of the ongoing missions. Figure 3 shows our team of UGVs.

3.3 Software

Three software platforms were used to task and control our team of UAVs and UGVs: MissionLab, ROCI, and Player/Stage.

MissionLab ([MissionLab, 2006]) is a suite of software tools for developing and testing behaviors for a single or team of robots. The user interacts through a design interface tool that permits the visualization of a specification as it is created. Individual icons correspond to behavioral task specifications, which can be created as needed or preferably reused from an existing repertoire available in the behavioral library. Multiple levels of abstraction are available, which can be targeted to the abilities of the designer, ranging from whole robot teams, down to the configuration description language for a particular behavior within a single robot, with the higher levels being those easiest to use by the average user. After the behavioral configuration is specified, the architecture and robot types are selected and compilation occurs, generating the robot executables. These can be run within the simulation environment provided by MissionLab itself for verification of user intent such as [Endo et al., 2004] and [MacKenzie and Arkin, 1998], or through a software-switch, that can be downloaded to the actual robots for execution.

ROCI ([Chaimowicz et al., 2003], [Cowley et al., 2004a]) is a software platform for programming, tasking and monitoring distributed teams of robots. ROCI applications are composed from self-describing components that allow for message-passing based parallelism that allows for the creation of robust, distributed software. ROCI is especially suited for programming and monitoring distributed ensembles of robots and sensors since modules can be transparently launched and connected across a network using mechanisms that provide automated data formatting, verification, logging, discovery, and optimized transfer.

Player is a device server that provides a flexible interface to a variety of sensors and actuators (e.g., robots). Player is language and platform independent allowing robot control programs to execute on any computer with network connectivity to the robot. In addition, Player supports multiple concurrent client connections to devices, creating new possibilities for distributed and collaborative sensing and control. Stage is a scaleable multiple robot simulator; it simulates a population of mobile robots moving in and sensing a two-dimensional environment, controlled through Player.

3.4 Communication

Every agent on the network is equipped with a small embedded computer with 802.11b wireless Ethernet call the junction box (JBox). Communication throughout the team and across the different software platforms was achieved via the wireless network. The JBox, developed jointly by the Space and Naval Warfare Systems Center, BBN Technologies, and the GRASP Lab, handles multi-hop routing in an ad-hoc wireless network and provides the full link state information enabling network connectivity awareness to every agent on the network.

4 Component Technologies

We present the component technologies developed towards the goal of providing an integrated framework for the command and control of an adaptive system of heterogeneous robots. These technologies were developed as a set of tools that can allow a human user to deploy a robot network to search and locate information in a physical world analogous to the use of computer networks via a search engine to look for and locate archived multimedia files. Of course, the analogy only goes so far. Unlike the World Wide Web, looking for a human target does not reduce to searching multimedia files that might contain semantic information about human targets. Robots must search the urban environment while keeping connectivity with a base station. They must be able to detect and identify the human target. And they must be able to alert the human operator by presenting information ordered in terms of salience, through a wireless network, allowing the human operator to request detailed information as necessary. Ideally, while all this is happening the process of reconfiguring, routing information through a multi-hop network, and moving to maintain connectivity must be transparent to the human user. In this section, we provide a brief summary of the enabling technologies developed to bring us closer to our vision. We refer the interested reader to the relevant literature for more detailed discussions.

4.1 Mission Specification and Execution

A pressing problem for robotics in general is how to provide an easy-to-use method for programming teams of robots, making these systems more accessible to the average user. The MissionLab mission specification system [MissionLab, 2006] has been developed to address such issue.

An agent-oriented philosophy [MacKenzie et al., 1997] is used as the underlying methodology, permitting the recursive formulation of entire societies of robots. A society is viewed as an agent consisting of a collection of either homogeneous or heterogeneous robots. Each individual robotic agent consists of assemblages of behaviors, coordinated in various ways. Temporal sequencing affords transitions between various behavioral states that are naturally represented as a finite state acceptor. Coordination of parallel behaviors can be accomplished via fusion, action-selection, priority, or other means as necessary. These individual behavioral assemblages consist of groups of primitive perceptual and motor behaviors, which ultimately are grounded in the physical sensors, and actuators of a robot. An important feature of MissionLab is the ability to delay binding to a particular behavioral architecture (e.g., schema-based [Arkin, 1998]) until after the desired mission behavior has been specified. Binding to a particular physical robot also occurs after specification, permitting the design to be both architecture- and robot-independent. This characteristic allowed the incorporation of the ROCI and Player/Stage systems.

To achieve the level of coordination required in an integrated mission involving a team of heterogeneous robots controlled by three different mobile software platforms (MissionLab, ROCI, and Player/Stage), the Command Description Language interpreter (CMDLi) was developed. The CMDLi is a common software library that is compiled into each of the software target platforms. It parses and executes a common text file (a CMDL script) that contains the integrated mission plan developed in MissionLab by the operator (see Figure 4(a)). Hence, the script has to be distributed among all the participating robots prior to execution. The CMDL script is organized into two parts (1) the background information and (2) a list of behavioral tasks to be executed sequentially. For example, the CMDL script used during the integrated experiment is shown in Figure 4(b). The background information includes the names of the robot executables and the information regarding memberships of predefined groups. At runtime, the CMDLi interpreter resident on each platform sequentially executes the list of specified behaviors, and sends corresponding commands to the underlying controller program (i.e., Player in Player/Stage, etc.).

Tasks/behaviors supported by CMDLi include *MoveTo, Loiter, TrackTarget*, *and Synchronize*. In *MoveTo*, the robot drives and steers itself towards the target position whereas, in *Loiter*, the robot stops and stands by at the target position. When the robot is executing the *TrackTarget* behavior, the robot identifies and follows a target object. In *Synchronize*, the robot waits for other specified robots to reach the same synchronization state. To realize this synchronization, each robot broadcasts its behavioral status to others via the JBox. When synchronization is attained, the robot resumes execution of the remaining mission.

The available tasks for CMDLi can be easily expanded to a more general list. Each of the three software platforms already supports various behavior repertoires. (For example, the robot controlled by MissionLab can execute more than 80 types of behaviors [MissionLab, 2006].) To add a new task to this CMDLi framework, simply a new binding between the new task name and the platform's behavior needs to be defined. It is important to note that increasing the size of the task list does not significantly affect computational complexity or performance as sequencing of the tasks is previously defined by a human operator rather than an automatic planning algorithm. Of course, if the new task involves a computationally very expensive algorithm (e.g., solving a traveling salesman problem), the performance should be solely affected by the execution of the task itself (i.e., the size of the list does not matter).

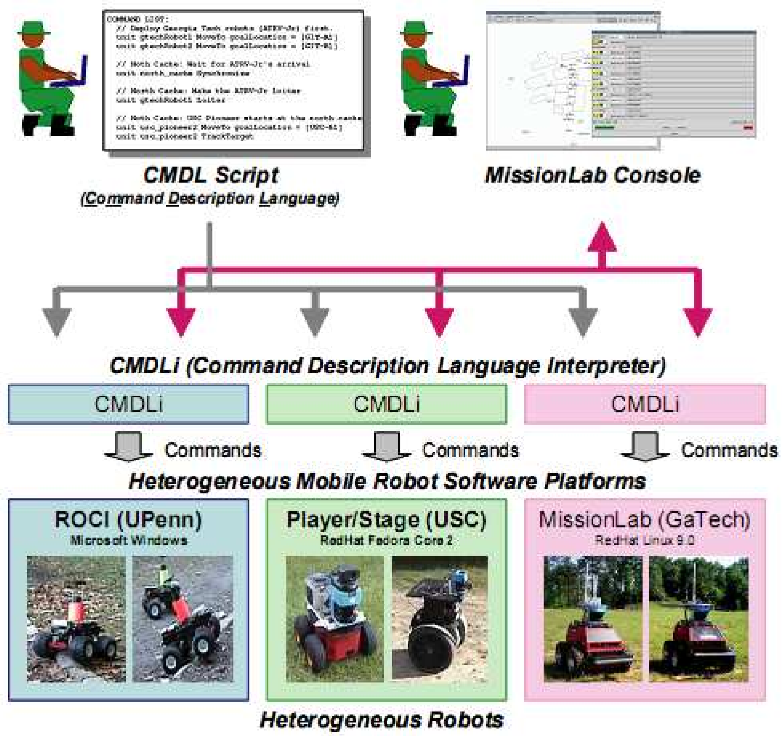
The status of the robot can also be monitored by the MissionLab console along with the overall progress of the mission. More specifically, the display consists of a mission area map showing the real-time GPS coordinates of the robots as well as a CMDLi interface that can dynamically display the progress of an integrated mission. A screen capture of the MissionLab console showing progress during the integrated experiment is depicted in Figure 5. In this particular example, at the North cache, ClodBuster 1 (controlled by ROCI and denoted by *upenn_1*) waits for ATRV-Jr 1 (controlled by MissionLab and denoted as gtechRobot1) to complete the *MoveTo* (GIT-A1) behavior, so that synchronization can be achieved. In the South cache, two ClodBusters (denoted by *upenn_2* and *upeen_3*), a Pioneer2 AT and a Segway (controlled by Player and denoted by *usc_pioneer*1 and *usc_segway* respectively) all wait for the second ATRV-Jr (denoted by *gtechRobot*2) to arrive at their cache.

Lastly, at any given point, the operator is given the option to interrupt or even abort the current mission via the CMDLi interface at the MissionLab console.

4.2 Communication network and control for communication

Successful deployment of multi-robot tasks for surveillance and search and rescue relies in large part on a reliable communication network. In general, radio propagation characteristics are difficult to predict a priori since they depend upon a variety of factors [Neskovic et al., 2000] which makes it difficult to design multi-agent systems such that the individual agents operate within reliable communication range at all times. In this section, we consider the problem of acquiring information for radio connectivity maps in urban terrains that can be used to plan multi-robot tasks and also serve as useful perceptual information.

A radio connectivity map is a function that returns the signal strength between any two positions in the environment. In general, it is extremely difficult to obtain a connectivity map for all pairs of positions in the desired workspace, thus one aims to construct a map for pairs of locations selected a priori. For small teams of robots, the construction of the radio connectivity map can be formulated as a graph exploration problem. Starting with an overhead surveillance picture, it is possible to automatically generate roadmaps for motion



(a)

```
MISSION NAME "Joint Demo"

NEW-ROBOT atrvjrClass "atrvjrRobot"
NEW-ROBOT truckClass "truckRobot"
NEW-ROBOT pioneerClass "pioneerRobot"
NEW-ROBOT segwayClass "segwayRobot"

UNIT <gtechRobot1> atrvjrClass
UNIT <gtechRobot2> atrvjrClass

UNIT <upenn_1> truckClass
UNIT <upenn_2> truckClass
UNIT <upenn_3> truckClass

UNIT <usc_pioneer1> pioneerClass
UNIT <usc_segway> segwayClass

UNIT <north_cache> (gtechRobot1 upenn_1)
UNIT <south_cache> (gtechRobot2 upenn_2 upenn_3
usc_pioneer1 usc_segway)
UNIT <all> (gtechRobot1 gtechRobot2 upenn_1 upenn_2
upenn_3 usc_pioneer1 usc_segway)

COMMAND LIST:
   // Deploy Georgia Tech robots (ATRV-Jr) first.
   unit gtechRobot1 MoveTo goalLocation = [GIT-A1]
   unit gtechRobot1 MoveTo goalLocation = [GIT-A2]
   unit gtechRobot1 MoveTo goalLocation = [GIT-A3]
   unit gtechRobot2 MoveTo goalLocation = [GIT-B1]
   unit gtechRobot2 MoveTo goalLocation = [GIT-B2]
   unit gtechRobot2 MoveTo goalLocation = [GIT-B3]

   // Noth Cache: Wait for ATRV-Jr's arrival
   unit north_cache Synchronize

   // North Cache: Make the ATRV-Jr loiter
   unit gtechRobot1 Loiter

   // Noth Cache: Send out the UPenn truck
   unit upenn_1 MoveTo goalLocation = [UPN-A1]
   unit upenn_1 MoveTo goalLocation = [UPN-A2a]
   unit upenn_1 MoveTo goalLocation = [UPN-A2b]
   //unit upenn_1 TrackTarget
   unit upenn_1 Loiter

   // South Cache: Wait for ATRV-Jr's arrival
   unit south_cache Synchronize

   // South Cache: Send out Upenn robots first.
   unit upenn_2 MoveTo goalLocation = [UPN-B1]
   unit upenn_3 MoveTo goalLocation = [UPN-C1]

   unit south_cache Synchronize

   // South Cache: Send out USC pioneer
   unit usc_segway MoveTo goalLocation = [USC-C1]

   unit south_cache Synchronize

   // South Cache: Do the rest
   unit gtechRobot2 Loiter
   unit usc_pioneer1 MoveTo goalLocation = [USC-A1]
   unit usc_pioneer1 Loiter
   unit upenn_2 MoveTo goalLocation = [UPN-B2a]
   unit upenn_2 MoveTo goalLocation = [UPN-B2b]
   unit upenn_2 Loiter
   unit upenn_3 MoveTo goalLocation = [UPN-C2a]
   unit upenn_3 MoveTo goalLocation = [UPN-C2b]
   unit upenn_3 Loiter
   unit usc_segway MoveTo goalLocation = [USC-C2]
   unit usc_segway MoveTo goalLocation = [USC-C1]
   unit usc_segway MoveTo goalLocation = [USC-C3]
   unit usc_segway TrackTarget
```

(b)

Figure 4: (a) Coordination of heterogeneous mobile robot software platforms through the command description language interpreter (CMDLi). (b) CMDL script used during the MARS 2020 integrated experiment.

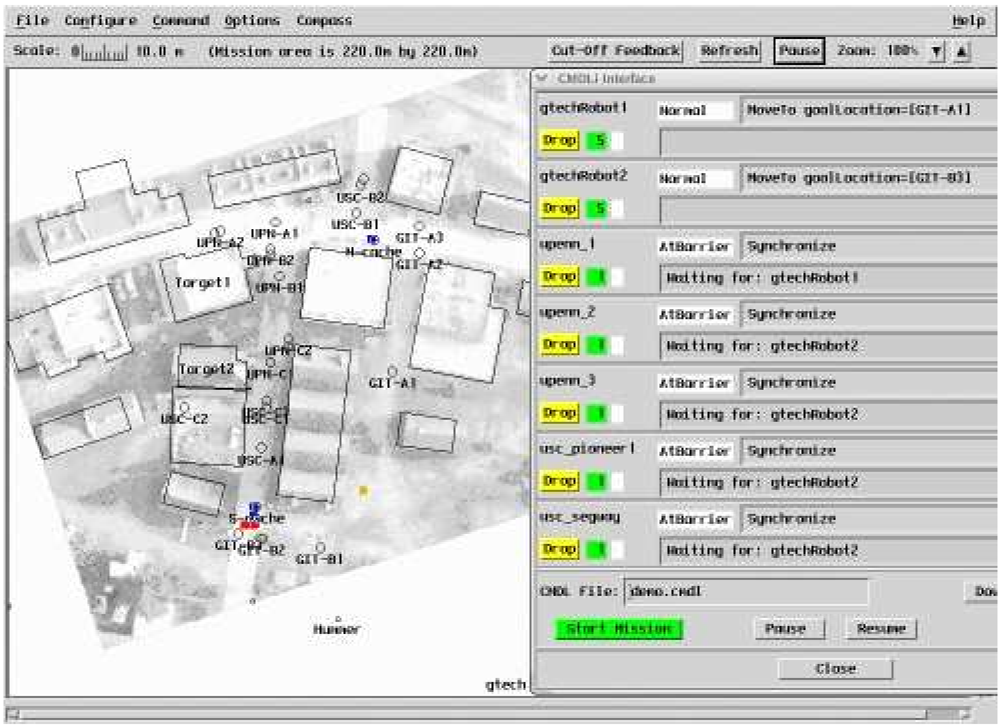


Figure 5: Screen capture of the MissionLab console showing the progress of the integrated mission. The locations of the robots with respect to the MOUT site map are displayed on the left-hand-side. The progress of the mission with respect to the CMDLi script is shown in the right-hand-side.

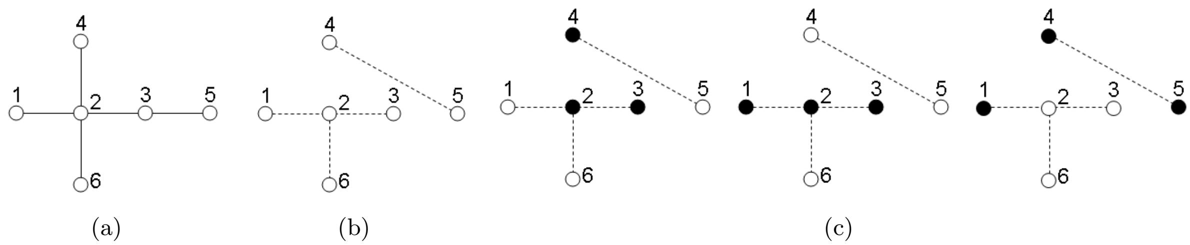


Figure 6: (a) A roadmap graph. The solid edges denote feasible paths between neighboring cells associated with each node. (b) A radiomap graph for (a). The dashed edges denote links for which signal strength information must be obtained. (c) Three sample configurations of three robots on the roadmap graph that can measure at least one of the edges in the radiomap graph. The solid vertices denote the location of each robot.

planning and encode these roadmaps as *roadmap graphs*[1]. From these roadmap graphs, a *radiomap graph* is obtained by determining the set of desired signal strength measurements (between pairs of positions) one would like to obtain. The discretization of the workspace allows us to strategically place each robot in a k-robot-team in k separate locations on the roadmap graph to obtain the desired measurements encoded in the radiomap graph. An example roadmap graph and its corresponding radiomap graph is shown in Figure 6. The solid edges in Figure 6(a) denote feasible paths between pairs of positions denoted by the circles. The dashed edges in Figure 6(b) denote signal strength measurements between pairs of positions that must be obtained. Figure 6(c) show three possible placements of a team of 3 robots such that the team can obtain at least one of the measurements given by the radiomap graph. An exploration strategy then consists of a set of waypoints each robot must traverse to obtain all the desired signal strength measurements encoded in the radiomap graph.

Experiments were performed using three of our ground vehicles to obtain radio connectivity data at the Ft. Benning MOUT site. In these experiments, an optimal exploration strategy was determined using the algorithm described in [Hsieh et al., 2004]. Each robot was individually tasked with the corresponding list of waypoints. Team members navigate to their designated waypoints and synchronize, every member of the team measures its signal strength to the rest of the team. Once the robots have completed the radio signal strength measurements, they synchronize before moving on to their next targeted location. This is repeated until every member has traversed through all the waypoints on their list. The waypoints are selected to minimize the probability of losing connectivity under line-of-sight conditions in the planning phase to ensure the success of the synchronization based on line-of-sight propagation characteristics that can be determined a priori. Figure 7 shows the radio connectivity map that was obtained for the MOUT site. The weights on the edges denote the average signal strength that was measured between the two locations. In these experiments, the signal strength was measured using the JBox, described in Section 3.

Radio connectivity maps can therefore be used to plan multi-robot tasks to increase the probability of a reliable communication network during the execution phase. Ideally, the measurements obtained during the exploration phase can be used to construct a limited

[1] In the event an overhead surveillance picture is not available, one can generate roadmaps for motion planning with a map of the region of interest

(a) (b)

Figure 7: (a) Overhead image of the MOUT site. (b) Experimental radio connectivity map for the MOUT site obtained using our multi-robot testbed.

model for radio propagation in the given environment such that, when coupled with additional reactive behaviors [Hsieh et al., 2006], a reliable communication network can be maintained during deployment. This two prong approach ensures that communication constraints are always satisfied and allows the operator to re-deploy the team and/or deploy additional assets in the presence of dynamic changes in the environment.

4.3 Programming abstractions and composition for multi-robot deployment

The software development process in robotics has been changing in recent years. Instead of developing monolithic programs for specific robots, engineers are using smaller software components to construct new, complex applications. Component based development offers several advantages such as reuse of code, increased robustness, modularity and maintainability. To this end, we have been developing ROCI, a software platform for programming, tasking and monitoring distributed teams of robots [Cowley et al., 2004a]. In ROCI, applications are built in a bottom-up fashion from basic components called ROCI modules. A module encapsulates a process which acts on data available on its inputs and presents its results as outputs. Modules are self-contained and reusable, thus complex tasks can be built by connecting the inputs and outputs of specific modules. We say that these modules create the language of the ROCI network, allowing task designers to abstract away low level details in order to focus on high level application semantics [Cowley et al., 2004b].

One key characteristic of a component-based system is the development of robust interfaces to connect individual modules. In component-based development, external interfaces should be clearly defined to allow an incremental and error resistant construction of complex applications from simpler, self-contained parts. By making interfaces explicit and relying on strongly-typed, self-describing data structures, ROCI allows the development of robust applications. Moreover, ROCI's modularity supports the creation of parallel data flows which favors the development of efficient distributed applications.

The composition of complex behaviors in a component-based system may be achieved through the use of a more declarative application specification that defines application components, parameters of those components, and the connections between components, as opposed to the more traditional imperative programming style the components themselves may be developed with. This delineates a separation between the specification of *what* an application does from *how* it does it. This division is enabled by the syntactic and semantic interface specifications associated with individual components, which may be generated automatically using type introspection or manually by the developers. The system should everywhere be designed to require minimal extra effort from the developer to support the notion of the actual distributed, compositional execution model.

The emphasis on interfaces further steers component development towards a natural implementation of message-passing parallelism, once again with minimal impact on the component developer. Indeed, the many pitfalls common to parallel processing should not be of primary concern to the developers of many types of modules whose behavior ought to be conceptually atomic. Instead, the application architect, working with the vocabulary defined by the component developers, may construct parallel data flows implicitly through the creation of a module network, the nature of which is of no intrinsic interest to the component developer.

4.4 Distributed databases for situational awareness

A data logging system has been built on top of the foundation described in the previous section as realized by the ROCI software platform. Due to the fact that component interfaces are defined in terms of the data types they transact, operations on component outputs may be automatically dispatched to an appropriate handler via traditional single dispatch. In this case, we developed a generic logging system that could maintain a store of data indexed by time. While the types of data relevant to a mobile robot deployment are varied, time is a universally meaningful index due to the sequential manner in which data is collected. This basic indexing can be augmented by additional mechanisms that handle more specific data types, for example indexing position measurements by location. These loggers operate independently of the components that generate the data, thus freeing the component developer from concerns regarding serialization, indexing, or query resolution. This functional separation is a hallmark of componentized development and is responsible for the extensibility of the system as a whole.

With these flexible data logging components in hand, an application over the robot network may be decorated with logs on any inter-component connection. These logs are then discoverable not just as generic data logs, but as data logs specific to the type of data they are attached to. This is made possible by the self-describing nature of inter-component connections based on the underlying type system. Having such logs attached to arbitrary data sources frees the development team from having to foresee every useful combination of sensor data. Instead, aggregate data types are created on-demand by cross-indexing separate data stores, perhaps across multiple machines. In this way, smart, compound data types are created from data streams that are annotated only with the metadata necessary to describe their own type; there is no unnecessary coupling imposed on the software architecture at any level.

The logging system itself was inspired by the observation that the sensor and processor bandwidth on-board many mobile robots far outstrips available bandwidth. Due to this imbalance, it is often beneficial to optimize query resolution over the distribution of data sources by distributing query logic to the data before performing a join over the results of that initial filtering step. In the ROCI system, it is easy to programmatically launch a component, or collection of components, on another machine and attach inputs and outputs to dynamically discovered data sources. The code of the component will be automatically downloaded by the node hosting the relevant data in question via a peer-to-peer search and download mechanism that is transparent to the node launching the component and the node that is to execute the component or collection of components. This allows for the creation of active queries that ship their logic to the data and return only resulting data sets to the originator of the query. In most usages, the result data set is significantly smaller than the data set taken as a whole.

An example of this functionality is the determination of where a particular target was sighted from. The query is a joining of a position table with an image table over the shared time index where the images contain a particular target. In our experimental setup, accurate position information was often logged by a camera system mounted on roof-tops overlooking the arena of operations, while the mobile robot logged many hundreds of megabytes of image data. The query, in this case, was executed by shipping a target identification component, parameterized to look for a specified target, to the node that maintained the image log. The time indices for images containing the target where used to index into the position log maintained by the node tracking the mobile units. Finally, the positions from which mobile units identified the target were sent to the query originator. Note that transferring the image data set over the network would be impractical; even transferring the position data set, which was generated from high-frequency sampling, would have been prohibitively expensive. Instead, resources were used commensurate with their availability.

4.5 Cooperative search, identification, and localization

In this section we describe the framework used to exploit the synergy between UAVs and UGVs to enable cooperative search, identification and localization of targets. In general, UAVs are adept at covering large areas searching for targets. However, sensors on UAVs are typically limited in their accuracy of localization of targets on the ground. On the other hand, UGVs are suitable for accurately locating ground targets but they do not have the ability to move rapidly and see through such obstacles as buildings or fences. Using the *Active Sensor Network* (ASN) architecture proposed in [Makarenko et al., 2004], we build upon the key idea that the value of a sensing action is marked by its associated reduction in uncertainty and that mutual information [Cover and Thomas, 1991] captures formally the utility of sensing actions in these terms. This allows us to incorporate the dependence of the utility on the robot and sensor state and actions and allows us to formulate the tasks of coverage, search and localization as optimal control problems. Our algorithms for search and localization are easily scalable to large numbers of UAVs and UGVs and transparent to the specificity of the individual platforms.

In this framework, the detection and estimation problems are formulated in terms of summation and propagation of formal information measures. We use *certainty grids* [Makarenko et al., 2003] as the representation for the search and coverage problems. The certainty grid is a discrete-state binary random field in which each element encodes the probability of the corresponding grid cell being in a particular state. For the feature detection problem, the state x of the i^{th} cell C_i can have one of two values, *target* and *no target*. This coverage algorithm allows us to identify cells that have an acceptably high probability of containing features or targets of interest.

The localization of features or targets problem is first posed as a linearized Gaussian estimation problem where the information form of the Kalman filter is used, [Grocholsky et al., 2003]. In this manner, one can show the influence of sensing processes on estimate uncertainty [Grocholsky et al., 2005] where the control objective is to reduce estimate uncertainty. Because this uncertainty directly depends on the system state and action, each vehicle chooses an action that results in a maximum increase in utility or the best reduction in the uncertainty. New actions lead to accumulation of information and change in overall utility. Thus local controllers are implemented on each robotic sensor platform that direct the vehicle and sensors according to the mutual information gradient with respect to the system state. This gradient controller allows individual vehicles to drive in directions that maximize their information gain locally. The additive structure of the update equations for the information filter lends itself to decentralization. Thus measurements from different robots (UAVs and UGVs) are propagated through the network and updated through propagation of inter-nodal information differences and decisions based on this updated information are made independently by each robot [Grocholsky et al., 2006]. A communications manager known as a *channel filter* implements this process at each inter-nodal connection [Grocholsky, 2002].

The network of aerial and ground sensor platforms can then be deployed to search for targets and for localization. Both the search and localization algorithms are driven by information-based utility measures and as such are independent of the source of the information, the specificity of the sensor obtaining the information, and the number of nodes that are engaged in these actions. Most importantly, these nodes automatically reconfigure themselves in this task. They are proactive in their ability to plan trajectories to yield maximum information instead of simply reacting to observations. Thus, we are able to realize a *proactive sensing network* with decentralized controllers, allowing each node to be seamlessly aware of the information accumulated by the entire team. Local controllers deploy resources accounting for and in turn influencing this collective information which results in coordinated sensing trajectories that transparently benefit from complementary sub-system characteristics. Information aggregation and source abstraction results in nodal storage, processing and communication requirements that are independent of the number of network nodes. The approach scales to large sensor platform teams.

4.6 Three-dimensional mapping

Many different methods can be used to represent outdoor environments. A point cloud [Wolf et al., 2005] is one of the most frequently used techniques. It can describe features in fine detail when a sufficient number of points is used. These maps can be generated fairly easily

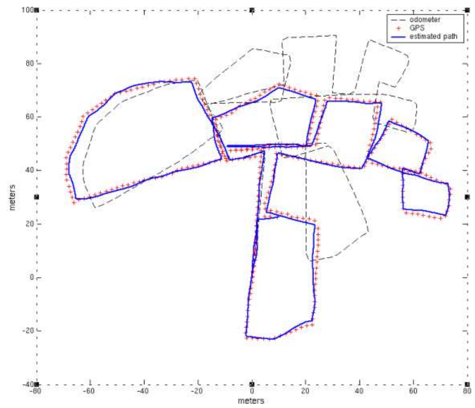


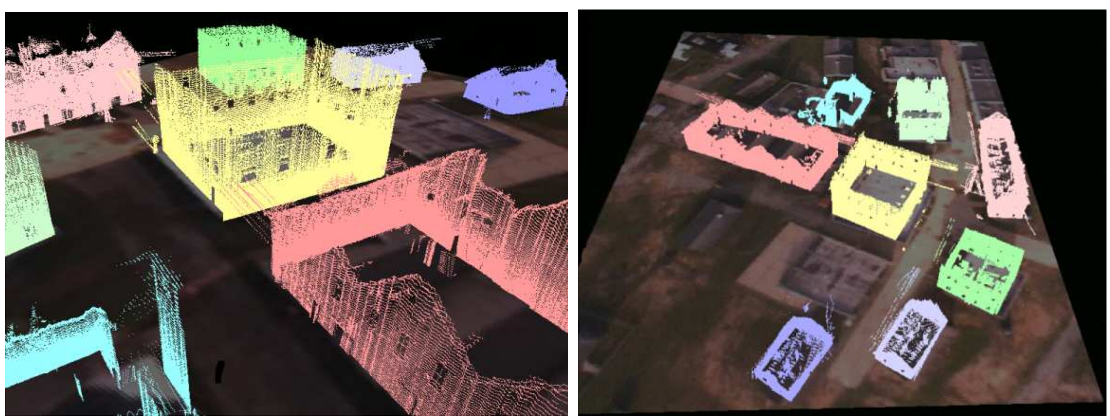
Figure 8: Localization on the MOUT site.

when good pose estimation and range information are available.

In order to smooth pose estimation, we developed a particle filter based GPS approximation algorithm [Wolf et al., 2005]. Each particle represents a possibility of the robot being at a determinate position, and the particles are propagated as the robot moves. The motion model for the particles is based on the odometer and IMU sensors data. A small amount of Gaussian noise is also added to compensate a possible drift in the robot's motion. The observation model is based on the GPS information. The particles are weighted based on how distant they are from the GPS points. Closer a particle is from the GPS point, higher it is weighted. After being weighted, the particles are re-sampled. The chance of a particle being selected for re-sampling is proportional to its weight; high weighted particles are replicated and low weighted particles are eliminated. The complete path of each particle is kept in the memory and at the end only particles that reasonably followed the GPS points will be alive. Consequently, the path of any of these particles can be used as reasonable trajectory estimation for the robot. The closer a particle is to the GPS point, the higher its probability for being selected. In order to obtain accurate local pose estimation, a scan matching algorithm is applied afterwards. Scan matching consists of computing the relative motion of the robot by maximizing the overlap of consecutive range scans. Features like trees, long grass, and moving entities make scan matching a hard task in outdoor environment. Figure 8 shows GPS data, odometry, and the particle filter-based GPS approximation for the robot's trajectory.

Once the current robot pose is obtained (from the localization module) and a desired target location/trajectory is specified, an VFH+ (Vector Field Histogram +) [Ulrich and Borenstein, 1998] algorithm is used for point-to-point navigation. VFH+ algorithm provides a natural way to combine a local occupancy grid map and the potential field method, and the dynamics and kinematics of a mobile robot can be integrated to generate an executable path. In addition, the robot's motion property (e.g. goal-oriented, energy-efficient, or smooth-path) can be controlled by changing the parameters of a cost function. Once the robot arrives at the desired way-point, the point-to-point navigation module notifies the achievement to CMDLi, and CMDLi proceeds to the next way-point. Figure 9 shows two trajectories that the robot generated while performing point-to-point navigation using two different way-point sets.

Figure 9: Point-to-point navigation using two way-point sets.



(a) (b)

Figure 10: Top and side view of the 3D map of Fort Benning site.

Thus, when constructing 3-dimensional maps based on the robot's position, the environment representation is built directly by plotting range measurements into the 3D Cartesian space. Figure 10 shows the result of mapping experiments performed at the Ft. Benning MOUT site. The maps were plotted using a standard VRML tool, which allows us to virtually navigate on the map. It is possible to virtually go on streets and get very close to features like cars and traffic signs and it is also possible to view the entire map from the top.

5 Integrated Demonstration

In this section, we describe the final experiment which demonstrated the integration of all the component technologies with a discussion of integration challenges that had to be overcome. In order to test and demonstrate the integration of the component technologies, we conceived an urban surveillance mission which involved the detection of a human target wearing a

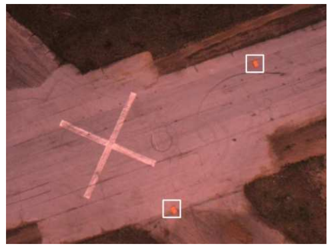


Figure 11: Targets localized by the UAV on the MOUT site encircled by a white square.

uniform with a specified color within a designated area, and then tracking the target once the identity of the target has been confirmed by a remotely located human operator. We briefly describe the mission in the next section that was used to stage a demonstration before discussing the results.

5.1 Demonstration Setup

To meet our project goals, an integrated demonstration based on an urban surveillance mission by a heterogeneous team of robots was conceived. The goal of the demonstration was for the team to ascertain if a human target with a particular uniform is within the surveillance region. The demonstration was conducted on December 1, 2004, at the Fort Benning MOUT site which approximately spans 90 meters North to South and 120 meters East to West. We deployed one UAV, three ClodBusters, two Pioneer2 ATs, two ATRV-Jrs and one Segway. Three human operators, responsible for monitoring the progress of the demonstration and target verification, were seated in the Hummer Vehicle which was used as the base station (Base). The experiment consisted of an aerial phase, where an UAV was tasked to conduct an initial coarse search of the surveillance region and determine potential target locations. This was then followed by a second phase, where UGVs, based on the UAV's initial assessment, were tasked to conduct more localized search and identification of the targets. Since the goal was surveillance rather than target recognition, targets in the aerial phase of the experiment consisted of bright orange color blobs and the target in the ground phase was a human in an orange colored vest. The orange color was simply used to ensure positive autonomous recognition without having to resort to complex and/or expensive means of target acquisition and designation. Furthermore, the human operator was brought into the loop on certain synchronization points. While deployment decisions (i.e. passing synchronization points) dedicated to maintaining network connectivity were made automatically, the human operator was engaged by the robots to confirm the identity of the target to ensure that the robots had indeed acquired the correct target before proceeding.

A single UAV was initially deployed to actively search and localize specified targets within the designated region. Targets were bright orange blobs located at various locations on the site. Once a target(s) was detected, an alert can then be sent from the UAV to the Base Station to trigger the deployment of the UGVs. Figure 11 show some targets detected by the UAV during one of these fly-by experiments.

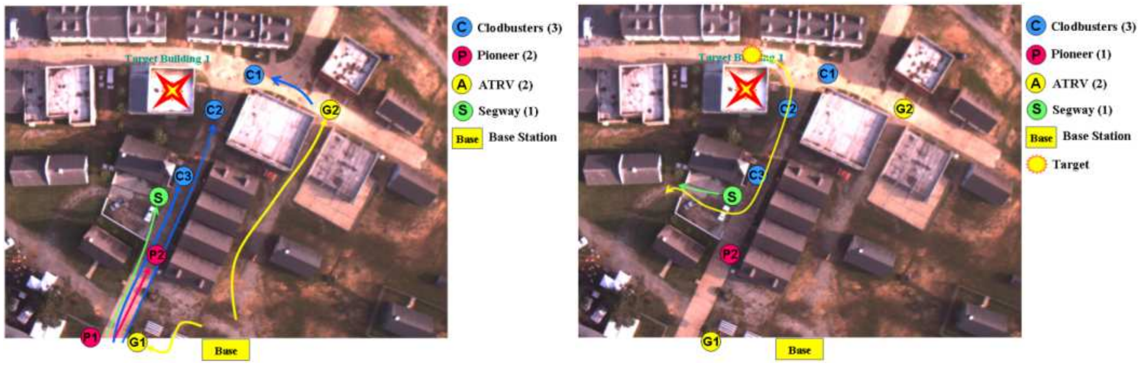
In our demonstration, we assumed a scenario where a UAV observed a human target entering

Figure 12: Robot initial positions and position of the Base.

the surveillance area from the north of the MOUT site which triggered an alert message at the base station. Once the Base had been notified, two groups of robots were dispatched from the Base to caching areas at the limits of radio network connectivity to await further instructions, marked as Cache N and Cache S in Figure 12. A ClodBuster was positioned at Cache N, while two ClodBusters, two Pioneer2 ATs, and a Segway were positioned at Cache S. The two ATRV-Jrs remained at the Base. For the ground phase of the experiment, the initial target sighting was selected a priori based on previous UAV experiments and thus the trigger was delivered manually.

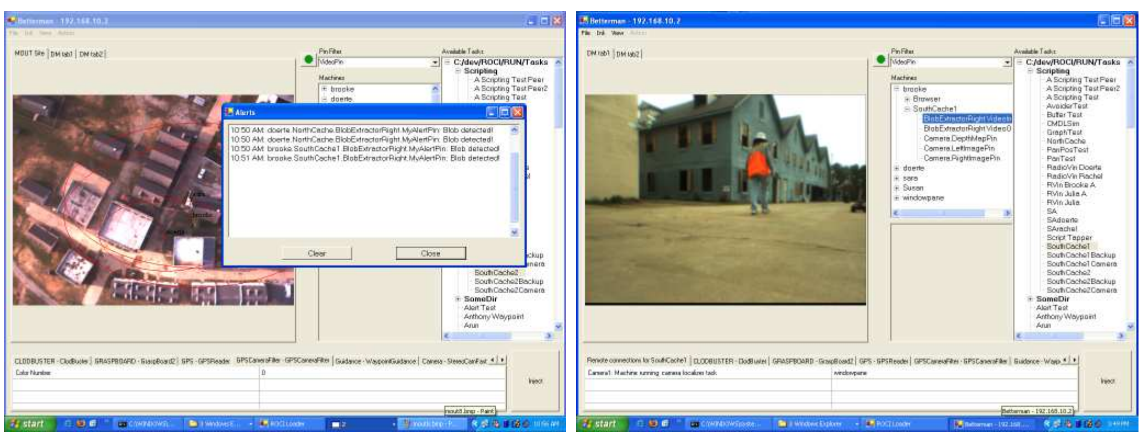
The human target then entered into the building, shown in Figure 12, unseen by the team. At this point, a surveillance mission was composed from the Base to search the town for the target of interest. The mission plan was initially given to two ATRV-Jrs which were then deployed, one to each cache area. Upon arrival, the mission plan was then transferred to the individual platforms, in this case already positioned at the two caches, via the wireless network. The two ATRV-Jrs then acted as radio network repeaters to allow the others to venture beyond the limit of one-hop network communication. Following a universal commence signal from the Hummer base station, the robots then autonomously deployed themselves to search for the target of interest. Once the ClodBuster robots had arrived at their target positions, they entered a scanning mode, and passed images of the candidate target to the operator. These positions were chosen during the mission planning phase based on the radio connectivity map of the MOUT site obtained during an initial mapping and exploration phase shown in Figure 7. A schematic of the deployment scheme and all the robot trajectories are shown in Figure 13(a). Network connectivity was maintained to ensure that once the target was located, an alert can be sent to the base station, permitting the operator to make a positive identification by viewing images obtained by the robotic agents. Figure 14(a) shows the actual alert that was seen by the human operator when the target was located by one of the ClodBusters. Figure 14(b) shows the image that was used by the human operator to make the positive identification. The individual robots autonomously selected the best image, i.e. images in which the target was centrally located, from their databases to forward to the Base when it was requested.

Once detected, the target was then tracked via the cameras from some of the robots, while the Segway moved in to physically track it as it left the area. This commitment to a particular target was finalized by the human, while the target tracking was achieved using a



(a) (b)

Figure 13: (a) Schematic of robot trajectories. (b) Schematic of target trajectory and Segway trajectory as it tracks and follows the target.



(a) (b)

Figure 14: (a) Screen capture of base station console showing the alert message notifying the human operator a target has been located. (b) Image obtained by one of the ClodBusters and provided to the human operator for identification.

(a) (b) (c)

Figure 15: Snapshots of previous target tracking result.

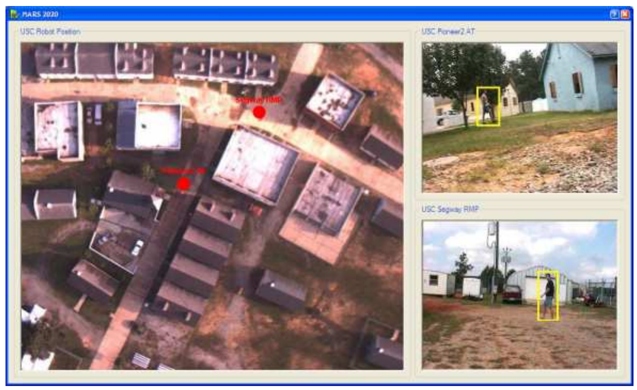


Figure 16: Screenshot of USC monitoring station.

particle filter-based algorithm developed to enable tracking in real-time [Jung and Sukhatme, 2004]. Figure 15 shows some snapshots of previous target tracking results. The information collected by the Segway was then transmitted to the base station over the multi-hop network. Figure 16 is a snapshot of the monitoring station. The current positions of the robots were displayed on the site map on the left in real-time. The two windows on the right showed live video streams from the Segway (on the top) and one of the Pioneer2 AT (on the bottom) for surveillance activity. Detected targets were displayed on top of the video streams.

The experiment concluded as the target of interest departed the bounds of the MOUT site, while the Segway tracked it movements. This experiment was carried out live and the deployment was fully autonomous with the experiment lasting approximately 30 minutes. A short movie of the integrated experiment has be included with this publication.

5.2 Challenges towards integration

5.2.1 Mission Specification and Execution

Specification and execution of a mission through MissionLab was found to be fairly robust. As the simulator in MissionLab allows the mission scenario to be tested without actual deployment of the robots, a solid CMDL script for the Ft. Benning MOUT site (100% success rate in simulation) was composed before being integrated with other components.

Even when the mission was executed by all of the actual robots, integrated with all other components, the CMDLi was found considerably reliable. Every robot was able to carry out all of the assigned tasks and the synchronization was properly attained as specified in the script. No major problem was found during the execution of the mission.

Improvements can be made to the mission specification process to enhance robustness to errors during execution. For example, during the demonstration, the Segway collided with one of the ClodBuster because of errors in localization (caused by poor GPS information) and because the Segway sensors for obstacle avoidance were not low enough to detect the smaller ClodBusters, it could have been prevented by explicitly modeling the heterogeneity of the robots and adding additional constraints on the waypoints of the individual robots.

5.2.2 Communication network and control for communication

A team of three ClodBuster robots were used to collect the radio signal strength map shown in Figure 7. The map was obtained prior to the final demonstration. Robots were simultaneously tasked to log signal strength and position information at specified location and continuously during the entire experiment. The continuous logs proved to be extremely useful in the construction of the map shown in Figure 7 since GPS errors of more than 5 meters were fairly common, especially towards the bottom right region of the site where robots consistently had problems obtaining accurate enough position information.

This experience suggests that it may be possible to obtain a finer resolution map if one can incorporate some learning into the exploration strategy. Additionally, while the map proved to be very useful for determining the deployment positions of the Clobuster robots in the final demonstration, it failed to provide much insight for the deployment positions of the other robots due to the differences in robot sizes and capabilities. Thus, future work includes the development of exploration strategies for teams of heterogeneous robots to enable better utilization of the various available resources. Lastly, since the robots were not required to operate at the limit of their communication links for the integrated demonstration, the radio connectivity map proved to be a useful resource. In situations where robots would be operating at these limits, one must incorporate reactive strategies to enable robots to adapt to changes in their signal strengths as shown in [Hsieh et al., 2006].

5.2.3 Programming abstractions and composition for multi-robot deployment

A benefit of componentized development is that it leads to a natural structuring of test activities. All components are extensively tested in isolation to ensure that they yield the proper results given some set of inputs. Unfortunately, the outputs of many components are not amenable to a binary classification of success or failure, and the input domains of many components are too large to provide total test coverage. A good example of the difficulty in component testing is a component designed to recognized a target object in images coming from a camera. Lighting conditions, distance to the target, and environmental features all have dramatic effects on the ability of the software to perform correctly. For this reason, such components were tested until they satisfied sometimes loosely-defined performance specification. In the MARS experiments, image components provided tuning parameters that let engineers adjust performance to match experimental operating conditions. Such tuning – used to account for lighting conditions in color-based segmentation and ground clutter in obstacle avoidance – presents a critical point of failure for the entire experiment. This weakness may be mitigated by self-tuning software that monitors its own performance when possible, and by tools designed to allow human operators to quickly calibrate the software when automated testing metrics are difficult to specify.

Isolation testing also presented a difficulty early in the development schedule when it was discovered that some components where designed with built-in performance assumptions. Such assumptions are only revealed when the component is tested in various execution environments, many of which may be difficult to anticipate without experimental experience. The simplest class of problems were those related to processing latency. Certain controllers were built with hard-coded real-time processing assumptions that could be satisfied by the host (non-real-time) OS under minimal CPU load, but violated when run alongside the many components that must coexist for the robot to be able to express all desired behaviors. Some of these controllers may be designed to dynamically tune parameters based on observed performance characteristics, while others may be moved to dedicated microcontrollers closer to the hardware. An approach that makes it easy to have asynchronous interoperation between multiple processors, some of which may be offering real-time performance guarantees, offers the engineering team the ability to avoid a potentially painful compromise, while also improving the scalability of the system as a whole.

5.2.4 Distributed databases for situational awareness

The distributed database for interactive situational awareness provision was successful on its own, but highly dependent on general robot performance. While the software could correctly task robots, and integrate disparate data stores into a cohesive view, there was little oversight of robot resources. This meant that a database query, such as a request for an image taken from a particular location, could fail due to the robot being unable to achieve its task due to a loss of GPS reception, a failure of the obstacle avoidance system, a mechanical failure, or a networking failure. All of these failure modes were observed, but there was no automated contingency management to handle such failures from a high level. A significant difficulty in designing such contingency plans is that each of these failure modes itself represents a possible outcome of myriad actual circumstances. An obvious strategy would be to attach a timeout to each query, and to send another robot if the first failed. Unfortunately, this strategy tended to result in either a robot pile-up at some environmental feature the robots were not capable of handling, or great inefficiency when the first robot was able to successfully recover and complete its mission. Ideally, such inefficiencies should be acceptable operating losses of a multi-robot system, and multiple robot losses could be an acceptable price for learning valuable information about the environment (i.e. do not send robots to this location), but our experimental setup was constrained by too small a population of robots to accept such eventualities.

5.2.5 Cooperative search, identification, and localization

In the demonstration, the synergy between aerial and ground vehicles was exploited to detect, identify and localize targets on the ground. Aerial vehicles are capable of searching quickly over a large area but they are unable to obtain accurate estimates of locations of potential targets because of errors in localization (position and orientation). On the other hand, ground vehicles are slower but capable of close-range observations that can confirm the identity of the target and provide better position information. This synergy was exploited during the demonstration. Over flight of the UAV (only one was flown for the demonstration) narrowed down the search area for the mission, while the ground robots were able to pursue potential targets for better information.

The integration of different communication networks and the distances involved proved to be challenging. The communication between the UAV and the base station involved a low bandwidth radio modem connection precluding the possibility of processing of data on the ground. Thus onboard image processing algorithms on an off-the-shelf laptop was necessary. The communication between the base station and the robots would have required multiple repeaters because of the distances between the base station and the robot. Instead of pursuing this solution we manually connected the UAV network and the UGV network allowing effective experimentation. A second challenge with cooperative behaviors with multiple sensors is the need to have an efficient data structure coding the information in the network that can be shared globally. While in principle this approach allows the scaling up to large numbers of anonymous vehicles, in practice, the communications manager needs to be aware of the identities of each vehicle to ensure there are no loops in the sensor fusion network. This is a research problem that is currently under investigation by several groups (see, for example, [Dellaert et al., 2005]).

5.2.6 Three-dimensional mapping

During our experiments in Ft. Benning, the robot mapped an area of approximately 50 m $\times$ 90 m (350 m tour with an average speed of 1.2 m/sec). A GPS unit (with accuracy of approximately 2m) was used as reference for the robot's pose estimation. The pose estimation error can be noticed in the walls of some buildings, which appear bent in the point cloud map. Unfortunately, ground truth was not provided during these experiments, but visual comparisons between the aerial image and the planar model suggest errors around 2 m, which is compatible with the GPS errors. A better reference for pose estimation would certainly lead our algorithm to generate more accurate models of the environemnt.

The performance can be further improved by extracting planar information from the incomplete point clouds. In our initial results, we represented flat surfaces found on point cloud map by planes [Wolf et al., 2005]. These planes do not possess the same level of detail as compared to the point clouds but they are more efficient in terms of memory. In situations were the application does not require a fine level of detail in the urban maps, planar information may be a convenient alternative.

In the future, we plan to investigate different methods for mapping urban environments and represent these maps efficiently even for large environments and high level of details. We are considering strategies for combining range information with images. Lastly, we are also considering the combination of range information provided by both ground robots and helicopters.

5.3 Analysis of Integrated Demonstration

Reliability and repeatability are most easily appreciated when viewed from a high level. One emphasis of the design of this demonstration was the importance of generality in the handling of robot resources. That is, the team was heterogeneous in make-up, and human operators should only be concerned with relevant data. The question of what robot provided what service should never come up, thus freeing human operators to focus on mission-specific goals, and offering welcome flexibility to engineering teams in terms of what robots are fielded for a given mission. This abstraction of hardware resources allows for a great level of scalability and fault tolerance.

During hardware warm-up at the start of the public iteration of the integrated demonstration, the GPS unit on one of the ClodBusters failed. The problem was quickly detected during the routine hardware start-up check (the importance of this activity having long been established over months of testing), but the device could not be made to respond. Given that there was no time to debug the issue, a quick solution was needed. In this case, the fastest fix was to pull the computer from the faulty robot, and plug it into a spare robotic platform that carried similar hardware. The swap worked smoothly, and no changes needed to be made to any other robot. Calibration settings for the camera and motors were adjusted to reflect the new hardware platform, but all high-level scripting remained unchanged. In this case, the abstraction of hardware resources to the scripting systems used to drive the demonstration provided great flexibility in terms of specific hardware provisioning.

The execution of the demonstration itself provided an example of the dangers of team heterogeneity, and the benefits of a loosely coupled robot team with built-in redundancy. The mission was specified such that three ClodBusters would be in position to view the person of interest as she left the building. Two robots had relatively close views of the two street-facing sides of the building, while a third robot had a slightly longer view that also encompassed a third side of the building. Once these robots were in position, the Segway was to begin patrolling an area adjacent to the building believed to house the target. This all proceeded according to plan, but, unnoticed by any observer, the third ClodBuster stopped slightly short of where it was intended to be positioned. Its location was within normal localization thresholds, but only just. Meanwhile, the Segway patrol also strayed towards the boundary of its expected area of coverage, which led to a run-in between the two robots, as previously mentioned. It was expected that the ClodBuster (GRASP) robots could have to contend with potential collisions among themselves, which they narrowly avoided during the demo, but such a collision could be prevented due to the robots detecting each other as unnavigable obstacles. The Segway, however, being a much larger robot, had its obstacle avoidance thresholds set such that an obstacle the size of a ClodBuster was considered to be surmountable. In this case, the Segway did surmount the GRASP robot soon after it sent its first long distance view of the target back to the Base. The GRASP robot was fairly seriously damaged, and began a spontaneous retreat back to its initial waypoint close to the Base. Fortunately, this third ClodBuster was not critical to mission success, and the remaining two GRASP vehicles were able to continue capturing images of the target to be selectively pushed to the human operator. The multiple views were sufficient for the human operator to positively identify the target, and allow the mission to proceed with the Segway tracking the target as she left the mission area. Had the mission hinged on a single ground vehicle to detect the target leaving a building in an urban setting, it would have been far more likely that positioning error or unwanted robot interaction (*i.e.* a collision) could have led to mission failure without any outright failure of any single component of the system.

6 Conclusion

Our vision for the demonstration was to advance the state-of-the-art in the integration of heterogeneous robots into a single team with minimal human intervention. This required the presentation of a single integrated command and control interface for the human operator that enabled him/her to task the team and monitor performance of the mission. This proved to be very challenging since the team consisted of diverse robotic assets from different universities, each running different operating systems and robot control architectures, and all quite different in physical size and capabilities.

Our final multi-robot coordination framework had to be both flexible and responsive for our team to be able to execute tasks efficiently and robustly. In our integrated demonstration at the McKenna MOUT site, the task was to patrol a small village and report and track a human target. The approach taken was to augment each robotic asset's controller with an instance of a distributed tasking software agent. For each robot, this agent negotiated work assignments with the other assets' agents and with the operator console to support assigning tasks across the assets. Each tasking agent instance maintained a work queue for its robot and passed commands and waypoints to the underlying controller for execution. It also aggregated status reported by the underlying controller and sent status reports back to the controller and to the other robots. This architecture allowed the operator to create a single mission for the team, distribute the mission to the robotic team members over the wireless network, and monitor, modify, or replace the mission during execution. In this fashion, the commander was able to deploy the mission across the team using the operator console and monitor progress of the mission and the location of vehicles on a map display during the demonstration. When a threat was identified the operator was presented with video of the potential target for confirmation.

Although the initial task assignment was centralized, control of the individual robotic assets was accomplished in a decentralized fashion so as to avoid the difficult task of seamless integration of all three command and control softwares. This strategy allowed team members to respond to dynamic changes in the environment as well as achieve full fault tolerance. Two of our robotic assets suffered catastrophic failures during mission deployment[2], however due to our decentralized architecture at the individual robot level, the team was still able to locate and track the target and complete the mission.

[2] One of the Pioneers failed to initialize while the Segway ran over one of the ClodBusters during the demonstration.

This experiment successfully demonstrated that diverse robots and robot control architectures could be reliably aggregated into a team with a single, uniform operator control station. It showed that disparate robots could perform tightly coordinated tasks, such as distributed surveillance and coordinated movements. Further, all of these capabilities were added as a software agent sitting on top of each robot's existing controller, without invasive modifications to the existing architecture or software.

Field-testing is expensive, tiring, and frustrating, but irreplaceable in moving the competency of the system forward. In the field, sensors and perceptual algorithms are pushed to their limits where vegetation, lighting, and terrain are uncontrollable, and communication radios struggle in cluttered areas with many nodes competing for bandwidth. Just ensuring each robot's batteries were charged at the same time to allow running an integrated experiment was difficult with this large a collection of robots. Much of the success of the integrated demonstration was due to the extensive testing of the individual subcomponents at each university and on the MOUT site.

Additionally, even with extensive field-testing, it is often difficult to guarantee system performance at execution time. Despite months of testing, GPS coverage was spotty at best during the final days leading up to the integrated experiment. To mitigate the localization problems, we placed stationary overhead camera nodes on key buildings on the MOUT site. These can be seen as the deployment of additional UAVs, capable of estimating their own positions as well as accurately track ground vehicles, deployed to provide localiztion support. Without this additional support, the integrated experiment would have failed due to localization problems. Success was dependent on our ability to anticipate and prepare for such failures.

As always, integration requires substantial planning and effort to be successful. This project, involving three universities and two corporations, benefited from strong leadership and collaboration to ensure that integration received the required emphasis and commitment from all involved.

Finally, the most important lesson was that bringing together the different groups into a single team was extremely beneficial and the whole was truly greater than the sum of the parts. Each team has unique capabilities that other teams could leverage to make rapid progress. Further, each style of robot has unique physical capabilities and sensors that were utilized to fill gaps and provide a solid collection of capabilities for the integrated team.

Acknowledgments

The authors would like to thank Dan Gomez-Ibanez, Joshua Karch and Rahul Swaminathan formerly from the University of Pennsylvania, Dr. Jason Redi and Keith Manning from BBN Technologies, and Dr. Tucker Balch, Yang Cheng, Zsolt Kira, Lilia Moshkina, Matt Powers, Alexander Stoytchev, Patrick Ulam, and Alan Wagner from the Georgia Tech Mobile Robot Laboratory. A special thanks to Mike Barnes and Irv (Rod) Rodriguez and their colleagues at the Soldier Battle Lab for their support at Ft. Benning. We gratefully acknowledge the support of DARPA MARS NBCH1020012.

References

Amarss (2006). Amarss - networked minirhizotron planning and initial deployment. http://research.cens.ucla.edu/.

Arkin, R. C. (1998). *Behavior-Based Robotics*. MIT Press.

Chaimowicz, L., Cowley, A., Sabella, V., and Taylor, C. J. (2003). Roci: A distributed framework for multi-robot perception and control. In *Proceedings of the 2003 IEEE/RJS International Conference on Intelligent Robots and Systems*, Las Vegas, NV.

Corke, P., Peterson, R., and Rus, D. (2003). Networked robots: Flying robot navigation using a sensor net. In *Proceedings of the Eleventh International Symposium of Robotics Research (ISRR)*, Springer Tracts on Advanced Robotics (STAR). Springer-Verlag.

Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. John Wiley.

Cowley, A., Hsu, H., and Taylor, C. J. (2004a). Distributed sensor databases for multi-robot teams. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA.

Cowley, A., Hsu, H., and Taylor, C. J. (2004b). Modular programming techniques for distributed computing tasks. In *Performance Metrics for Intelligent Systems (PerMIS) Workshop*, Gaithersburg, MD.

Dellaert, F., Kipp, A., and Krauthausen, P. (2005). A multifrontal qr factorization approach to distributed inference applied to multi-robot localization and mapping. In *Proceedings of the National Conference on Artificial Intelligence (AAAI '05)*, volume 5, pages 1261–1266, Pittsburgh, PA, USA.

Durrant-Whyte, H., Stevens, M., and Nettleton, E. (2001). Data fusion in decentralised sensing networks. In *Proc. Fusion 2001*, pages 302–307, Montreal, Canada.

et. al., L. C. (2005). Deploying air-ground multi-robot teams in urban environments. In *Proceedings of the 2005 International Workshop on Multi-Robot Systems*.

Endo, Y., MacKenzie, D. C., and Arkin, R. C. (2004). Usability evaluation of high-level user assistance for robot mission specification. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(2).

Fox, D., Ko, J., Konolige, K., Limketkai, B., Schulzz, D., and Stewart, B. (2006). Distributed multi-robot exploration and mapping. In *Proceedings of the IEEE*.

Grocholsky, B. (2002). *Information-Theoretic Control of Multiple Sensor Platforms*. PhD thesis, The University of Sydney. Available from http://www.acfr.usyd.edu.au.

Grocholsky, B., Keller, J., Kumar, V., and Pappas, G. J. (2006). Cooperative air and ground surveillance: Scalable approach to detection and localization of targets by a network of uavs and u. *IEEE Robotics and Automation Magazine*, 13(3):16–25.

Grocholsky, B., Makarenko, A., Kaupp, T., and Durrant-Whyte, H. (2003). Scalable control of decentralised sensor platforms. In *Information Processing in Sensor Networks: 2nd Int Workshop, IPSN03*, pages 96–112.

Grocholsky, B., Swaminathan, R., Keller, J. F., Kumar, V., and Pappas, G. (2005). Information driven coordinated air-ground proactive sensing. In *IEEE International Conference on Robotics and Automation*, Barcelona, Spain.

Howard, A., Parker, L. E., and Sukhatme, G. S. (2006). Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. In *International Journal of Robotics Research*, number 25 in 5-6, pages 431–447.

Hsieh, M. A., Cowley, A., Kumar, V., and Taylor, C. J. (2006). Towards the deployment of a mobile robot network with end-to-end performance guarantees. In *International Conference on Robotics and Automation (ICRA) 2006*, Orlando, FL.

Hsieh, M. A., Kumar, V., and Taylor, C. J. (2004). Constructing radio signal strength maps with multiple robots. In *Proc. IEEE International Conference on Robotics and Automation*, New Orleans, LA.

Jones, E., Browning, B., Dias, M. B., Argall, B., Veloso, M., and Stentz, A. T. (2006a). Dynamically formed heterogeneous robot teams performing tightly-coordinated tasks. In *International Conference on Robotics and Automation*.

Jones, E., Dias, M. B., and Stentz, A. T. (2006b). Learning-enhanced market-based task allocation for disaster response. Technical Report CMU-RI-TR-06-48, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Jung, B. and Sukhatme, G. S. (2004). Detecting moving objects using a single camera on a mobile robot in an outdoor environment. In *International Conference on Intelligent Autonomous Systems (IAS)*, pages 980–987, Amsterdam, The Netherlands.

Kaiser, W., Pottie, G., Srivastava, M., Sukhatme, G. S., Villasenor, J., and Estrin., D. (2005). Networked infomechanical systems (nims) for ambient intelligence. In *Ambient Intelligence*, pages 83–114. Springer.

Kang, W., Xi, N., and Spark, A. (2000). Formation control of autonomous agents in 3d workspace. In *Proc. IEEE International Conference on Robotics and Automoation ICRA'00*, pages 1755–1760, San Francisco, CA.

Kotay, K., Peterson, R., and Rus, D. (2005). Experiments with robots and sensor networks for mapping and navigation. In *International Conference on Field and Service Robotics*, Port Douglas, Australia.

Lerman, K., Jones, C., Galstyan, A., and Mataric, M. J. (2006). Analysis of dynamic task allocation in multi-robot systems. *International Journal of Robotics Research*.

MacKenzie, D. C. and Arkin, R. C. (1998). Evaluating the usability of robot programming toolsets. *International Journal of Robotics Research*, 4(7).

MacKenzie, D. C., Arkin, R. C., and Cameron, J. (1997). Multiagent mission specification and execution. *Autonomous Robot*, 4.

Majumder, S., Scheding, S., and Durrant-Whyte, H. (2001). Multi sensor data fusion for underwater navigation. *Robotics and Autonomous Systems*, 35(1):97–108.

Makarenko, A., Brooks, A., Williams, S., Durrant-Whyte, H., and Grocholsky, B. (2004). An architecture for decentralized active sensor networks. In *IEEE International Conference on Robotics and Automation (ICRA'04)*, New Orleans, LA, USA.

Makarenko, A., Williams, S., and Durrant-Whyte, H. (2003). Decentralized certainty grid maps. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3258–3263.

Mataric, M. J., Sukhatme, G. S., and Ostergaard, E. (2003). Multi-robot task allocation in uncertain environments. *Autonomous Robots*.

McMillen, C. and Veloso, M. (2006). Distributed, play-based role assignment for robot teams in dynamic environments. In *Proceedings of DARS-2006*, Minneapolis, MN.

MissionLab (2006). Missionlab: User manual for missionlab 7.0. College of Computing, Georgia Institute of Technology.

Neskovic, A., Neskovic, N., and Paunovic, G. (2000). Modern approaches in modeling of mobile radio systems propagation environment. *IEEE Communications Surveys*, Third Quarter.

Parker, L. E. (2005). Alliance: An architecture for fault tolerant multi-robot cooperation. In *Proc. IEEE International Conference on Robotics and Automation*, Barcelona, Spain.

Rybski, P. E., Stoeter, S. A., Erickson, M. D., Gini, M., Hougen, D. F., and Papanikolopoulos, N. (2000). A team of robotic agents for surveillance. In *Agents'2000*, Barcelona, Spain.

Spletzer, J. and Taylor, C. J. (2002). Sensor planning and control in a dynamic environment. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC.

Stroupe, A. and Balch, T. (2003). Value-based observation with robot teams (vbort) using probabilistic techniques. In *Proceedings International Conference on Advanced Robotics (ICAR)*, Portugal.

Sukhatme, G. S., Dhariwal, A., Zhang, B., Oberg, C., Stauffer, B., and Caron, D. A. (2006). The design and development of a wireless robotic networked aquatic microbial observing system. In *Environmental Engineering Science*.

Sukkarieh, S., Nettleton, E., Grocholsky, B., and Durrant-Whyte, H. (2003). *Information Fusion and Control for Multiple UAVS*. Klewer.

Tang, F. and Parker, L. E. (2005). Asymtre: Automated synthesis of multi-robot task solutions through software reconfiguration. In *Proc. IEEE International Conference on Robotics and Automation*, Barcelona, Spain.

Thibodeau, B. J., Fagg, A. H., and Levine, B. N. (2004). Signal strength coordination for cooperative mapping. Technical report, Department of Computer Science, University of Massachusetts, Amherst.

Ulrich, I. and Borenstein, J. (1998). Vfh+: Reliable obstacle avoidance for fast mobile robots. In *Proceeding of the IEEE International Conference on Robotics and Automation*, pages 1572–1577, Leuven, Belgium.

Vaglienti, B. and Hoag, R. (2003). *Piccolo system user guide*. Available from http://www.cloudcaptech.com/downloads.htm.

Wolf, D. F., Howard, A., and Sukhatme, G. S. (2005). Towards geometric 3d mapping of outdoor environments using mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1258–1263.