In Autonomous Robots and Multirobot Systems (ARMS) Workshop at Autonomous Agents and Multiagent Systems (AAMAS), Taipei, Taiwan, May 2011.

Designing the HRTeam Framework: Lessons Learned from a Rough-and-Ready Human/Multi-Robot Team

Elizabeth Sklar[1,4], A. Tuna Ozgelen[1,4], J. Pablo Munoz[1], Joel Gonzalez[2], Mark Manashirov[1], Susan L. Epstein[3,4], and Simon Parsons[1,4]

[1] Brooklyn College, The City University of New York, USA
[2] City College, The City University of New York, USA
[3] Hunter College, The City University of New York, USA
[4] The Graduate Center, The City University of New York, USA
corresponding author: sklar@sci.brooklyn.cuny.edu

Abstract. In this workshop paper, we share the design and on-going implementation of our HRTeam framework, which is constructed to support multiple robots working with a human operator in a dynamic environment. The team is comprised of one human plus a heterogeneous set of inexpensive, limited-function robots. Although each individual robot has restricted mobility and sensing capabilities, together the team members constitute a multi-function, multi-robot facility. We describe low-level system architecture details and explain how we have integrated a popular robotic control and simulation environment into our framework to support application of multi-agent techniques in a hardware-based environment. We highlight lessons learned regarding the integration of multiple varying robot platforms into our system, from both hardware and software perspectives. Our aim is to generate discussion amongst multi-robot researchers concerning issues that are of particular interest and present particular difficulties to the multi-robot systems community.

1 Introduction

This paper reports on the design and on-going implementation of a framework to support experimentation with mixed-initiative human/multi-robot teams. Our *HRTeam* framework is constructed to support multiple robots working with a human operator in a dynamic, real-time environment. The team is comprised of one human (the *operator*) plus a heterogeneous set of inexpensive, limited-function robots. Although each individual robot has restricted mobility and sensing capabilities, together the team members constitute a multi-function, multi-robot facility. The robots can be controlled directly by the human operator, or they can operate autonomously, without needing to wait for tele-operator input. Control of the robots is shared between the human operator and a software controller, and the locus of control can switch during run-time. The research questions we are investigating center around issues well-studied in the (virtual) *Multi-Agent Systems (MAS)* community. We are interested in how to coordinate activity and

allocate tasks to team members in a real-time, dynamic environment. We are also interested in how to integrate input from the human operator so that she is neither overwhelmed (because too much input is required) or consulted too rarely (so that overall task completion suffers). These issues present particular difficulties to the *Multi-Robot Systems (MRS)* community. Finding ways to address them is the focus of discussion here.

Our research is motivated by two related application areas: *urban search and rescue* [48, 67, 102] and *humanitarian de-mining* [38, 82]. In both instances, teams of robots are deployed to explore terrain that is potentially unsafe for humans and to locate targets of interest. In the first case, robots explore an enclosed space, such as a collapsed building, and search for human victims who may be physically trapped. The goal is to locate these victims and transmit their positions to human operators, so that human first responders can remove the victims to safety. In the second case, robots explore an open space, such as a field in a war zone, to search for anti-personnel mines that may be hidden from view. The goal is to locate these mines and transmit their positions to human operators, so that the mines can be disarmed and the area rendered safe for people to traverse.

Both application areas have a number of fundamental tasks in common. First, a robot must be able to explore a region (traverse and maneuver in the physical space) and *localize* (determine and track its position there). Second, a robot must be able to *recognize* objects of interest, using on-board sensors and possibly augmented intelligence to interpret sensor input. Third, a human operator must be able to communicate with the robots remotely and *strategize* so that the team can accomplish its overall task effectively. Ideally, in such a collaborative system, the human operator should not be overloaded with tasks, and the robots should not be idle. The team members should work together to accomplish the team's goal(s), taking advantage of members' individual abilities and strengths to complete tasks effectively and efficiently. Strategies to address these issues often stem from the MAS solutions implemented in virtual environments—where agents can have perfect and often complete information. Unfortunately, in a multi-robot setting, most information is noisy, incomplete, and often out-of-date. The challenge is to identify which MAS solutions can work in an MRS environment and adapt them accordingly.

As with any robotics research, a substantial effort must be made on the engineering side before any of these research questions can be investigated fully or satisfactorily. These efforts are more challenging in a multi-robot environment, simply because there are more hardware issues to contend with. Further, in a heterogeneous multi-robot environment, solving hardware problems for one (class of) robot does not necessarily solve the same problems for another (class of) robot; indeed, sometimes fixing one can break another. Finally, because we restrict our choice of hardware to inexpensive, limited-function robot platforms, additional constraints are presented. Note that this last aspect is not purely a function of budgetary realities, but rather part of our philosophy. There are always issues that arise when transferring results from a research environment

to a real-world setting. Often these issues are of a practical nature, for example, network interference or uneven lighting conditions that did not occur in the lab suddenly confront a system deployed in a new venue. Practicalities can render elegant laboratory solutions useless outside the lab. By creating a less-than-ideal lab environment, we hope to be address some of these practical issues in our everyday setting.

In this workshop paper, we share the design of our HRTeam framework. We describe low-level system architecture details and explain how we have integrated a popular robotic control and simulation environment (Player/Stage [35, 97]) into our framework to support application of multi-agent techniques in a hardware-based environment. We highlight lessons learned regarding the integration of multiple varying robot platforms into our system, from both hardware and software perspectives. Our aim is to generate discussion amongst multi-robot researchers concerning issues that are of particular interest and present particular difficulties to the MRS community. Finally, we close with a brief summary and status report on our ongoing research investigations.

2 Related work

Research on Multi-Robot Systems, where more than one mobile robot is used, considers challenges faced by individual robots and how a robot team might help address these challenges. Areas of investigation include localization [23, 28, 71], mapping and exploration [19, 89], and strategies to manage wireless connectivity among robots [77]. With simultaneous localization and mapping (SLAM) [3, 39, 46, 94], additional information from several robots can simplify a problem and speed the solution that would have been provided by a single robot [28]; although multi-robot SLAM can also lead to inconsistency in position estimates [47, 58]. Other challenges for a multi-robot team are similar to those for one robot, complicated by the need to merge or expand single-robot solutions to incorporate other robots. Path planning [2, 11, 57, 93] is one well-studied example of this. Another example is the learning of controllers for teams of robots [69, 70], which is more complex than learning for individual robots.

The largest category of work on multi-robot systems, however, cannot be compared with work on single robots. Some tasks cannot be accomplished by one robot, such as the transport of an object too large for a single robot to move [24, 76, 91, 101]. Other issues, such as the dynamic allocation of tasks to robots [4, 5, 16, 60, 64, 87, 96], simply do not arise with a single robot. *Task allocation* is particularly challenging and has received substantial attention. The distribution of responsibilities among a group of individuals is a complex optimization problem. It is made more difficult because robot team requirements change over time [96], and because the abilities of individual robots to address particular tasks are conditioned on their changing locations. Heterogeneous robot teams, where each member has different capabilities, further complicate the optimization problem.

The task allocation literature for multi-robot teams includes a strong thread on the use of auctions [32, 54, 56, 83] and market-based mechanisms in general

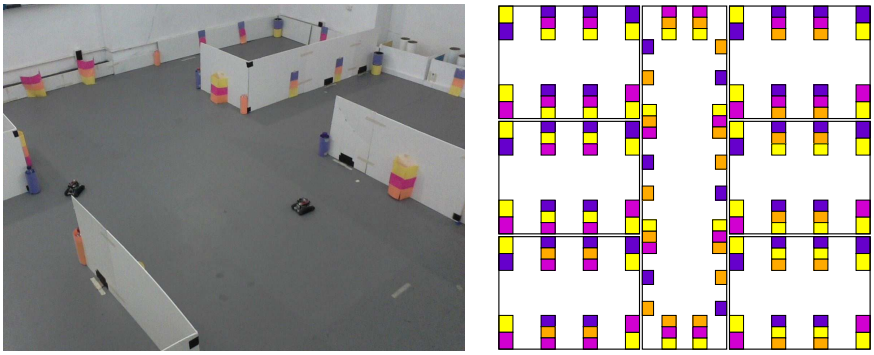
[20, 22, 33, 34, 103]. This work offers the various tasks for "sale" to robot team members. Individual robots indicate how much they are willing to "pay" to obtain tasks, and tasks are allocated based on bids for them—typically to the robot that makes the best offer. For example, this approach has been used to organize robots for exploration tasks [50, 51, 104]. Areas to explore were offered "for sale," and robots bid based on their distance to the locations on offer. Allocation favored lower bids, and thereby tended to allocate areas closer to robots. The market was constructed, however, to ensure that robots did not remain idle when several robots were initially close to the same unexplored area. Another example is the use of simple auctions to allocate roles, and correspondingly, tasks associated with those roles, to robots on a multi-robot soccer team [30, 31]. Robots "bid" on roles based on their proximity to the ball and the goal. Roles changed in real time, as the game progressed. The ability both to consider individuals' changing abilities and to balance those against the performance of a team as a whole makes market-based approaches attractive.

Early work on multi-robot systems [14, 73] included *foraging*, a standard task that had robots systematically sweep an area as they searched for objects (e.g., [61, 59]). This has much in common with search and rescue, and with humanitarian demining—our target areas of application. Techniques have been developed to ensure that the entire boundary of a space is visited [99], that search finds a specific target [6, 7, 41, 42, 80, 84], that a mobile target is kept under constant observation [75, 68], and that a human-robot team can exchange search roles flexibly and appropriately [43].

Finally, given our focus on the deployment of many small robots, we should mention work on swarm robotics [88, 61]. Though recent work on swarms has looked at more focused task allocation to different robots [63] and on ensuring that the swarm spreads across different target sites [40], work in this area differs from ours in by being less deliberative, relying on numbers and randomness to get coverage rather than thoughtful deployment of resources and in only dealing with homogeneous collections of robots.

Human-Robot Interaction (HRI) supports collaborative activities by humans and robots to achieve shared goals. Typical HRI research concentrates on the development of software and/or hardware to facilitate a wide range of tasks. These include robots maneuvering in physical spaces, both those designed for humans (e.g., [53]) or unfit for humans (e.g., [66]); people programming complex robots (e.g., [81]) or different types of simple robots (e.g., [8]); robots cooperating with human partners (e.g., [12, 25, 86, 98, 100]) and with other robots (e.g., [21, 55, 62, 92]); and user interfaces for communicating with robots (e.g., [49, 79]). Deployed HRI applications include cleaning [78], helping the elderly [95], assisting first responders in search and rescue tasks [17], demining in military settings [29], and teaching [52].

There are three main categories of control architectures for human-robot systems [37]: *fully autonomous*, where robots make decisions and control their actions on their own; *directly controlled*, where robots are driven by human operators; and *mixed-initiative* [15, 45], where robots share decision making with



(a) view of test arena (b) schematic of arena landmarks

Fig. 1. Robots' physical environment

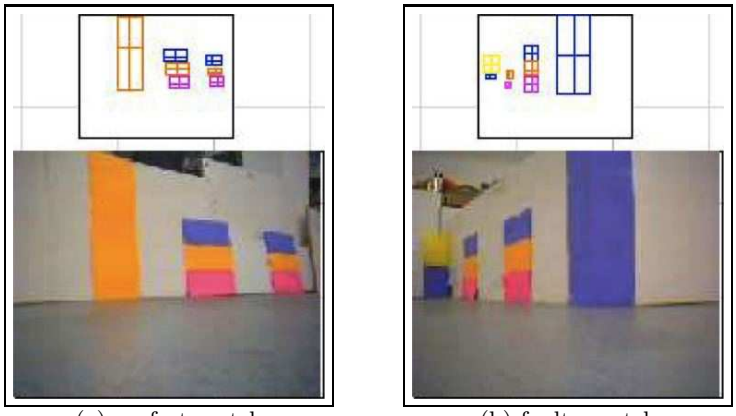
human users. Mixed-initiative systems reflect recent trends within the HRI community toward *socially intelligent* interfaces [9, 10, 26, 18] in which the aim is for robots and humans to respond to each other naturally. We highlight several mixed-initiative approaches here. *Adjustable autonomy* in a human-robot system permits dynamic transfer of control from human to robot and vice versa (e.g., [36, 85]). *Collaborative control* offers a dialog-based architecture in which decisions are "discussed" and made in real-time (e.g., [27]). Other examples of mixed-initiative systems include an affect-based architecture [1], and statistical techniques to infer missing information in human-robot communication [44].

We see our work as being within adjustable autonomy. Our first major research goal is to establish how best to transition control of a robot from a human to the robot and, especially, back again. (With a large robot team the human operator must be used sparingly to avoid overload). Our second major researcg goal is to investigate how best to coordinate the robot team when it is operating autonomously. With regard to this latter aim, we plan to test a range of coordination techniques from the multiagent systems literature, taking techniques that have been tested theoretically and in simulation, and seeing how they perform in the rough-and-ready world of robotics.

3 Physical Environment

Our test arena, shown in Figure 1a, is a physical environment that is divided into seven regions: six rooms and a hallway. Each region contains color-coded landmarks to guide the robots using vision-based localization. Figure 1b contains a schematic of the landmarks[1]. These are composed of vertically-aligned markers

[1] Note that the landmarks are a proxy for more sophisticated vision processing that would allow us to recognise unique features of the test arena. Using the landmarks allows us to test other aspects of our environment as we develop this vision capability.



(a) perfect match (b) faulty match

Fig. 2. Landmark identification

with stacked bands of one, two, or three colors. The entire color palette consists of four colors: yellow, pink, orange, and purple. On the northeast corner of each of the six rooms, a "purple-over-yellow" landmark is placed. The northwest corner contains a "yellow-over-purple" landmark; the southwest corner contains "yellow-over-pink"; and the southeast corner contains "pink-over-yellow". Inside each room, a unique 3-color marker distinguishes that room from the others; each of the room markers includes a purple band. In the hallway, a set of 3-color markers (without purple bands), using four unique color band permutations, mark the north, west, south and east walls of the hallway. Inside the hallway, the entrance to each room is marked with a single-colored purple landmark on the right side of the "doorway", and an orange landmark on the left.

The lighting conditions in the arena vary from one room to another. This means that it is not possible to have a single, non-overlapping color map with which to calibrate the colored landmarks; e.g., the orange and yellow color ranges tend to bleed together in some parts of the arena. The process of identifying landmarks involves first capturing images with robots' cameras and analyzing the images for "blobs" of color, then the color blobs are matched with landmarks from a dictionary of known objects. An example is shown in Figure 2. The figure on the left shows a perfect match between a robot's image and the markers that were identified. The figure on the right, however, has missed one blob of color (a purple band at the top of the second marker from the left), which makes it difficult to identify that marker correctly. Some of our research involves applying machine learning techniques to a participatory human/robot process in which the system learns a reliability metric for the images with help from the human operator. While the system can recognize that problems exist without the help

The large number of landmarks are required because of the fixed cameras used by most of the robot platforms.

(a) AIBO (b) Create (c) Fribbler (d) NXT (e) SRV-1

Fig. 3. Robot gallery

of a human, having a human in the loop can speed the learning process. In the example shown in Figure 2b, the system can quickly detect a problem with the image simply because there are no markers in its dictionary that consist of only an orange band above a pink band.

As mentioned earlier, the robots on our team are inexpensive, limited-function platforms. These are pictured in Figure 3. We have been experimenting with five different platforms, spanning a range of sensing and locomotion capabilities and communication technologies. Table 1 lists the hardware differences. Only the AIBO has a powerful enough on-board processor to function as a stand-alone platform. The Create is mounted with a Hokuyo URG-04LX Scanning Laser Rangefinder and a Dell laptop that communicates, via USB, to the robot and the laser device. The Fribbler and the SRV-1 have minimal on-board memory and so are controlled by off-board laptops with dedicated communication channels. The NXT has limited on-board memory and processing capabilities—more than the Fribbler and SRV-1, but substantially less than the AIBO. Currently, we operate the NXT in the same way as the Fribbler and SRV-1: via off-board laptop with dedicated communication channel. All of the devices listed as "wireless" in Table 1 use 802.11. The SRV-1 platform was originally built using an XBee radio device. Newer "Blackfin" models are now available with 802.11. We have found that the XBee radio suffers greatly from interference with the 802.11, particularly when the two types of communicating devices are in close proximity with one another. We have also found that we must make judicious use of 802.11 communication, otherwise it is quite easy to flood our local network—for example, when multiple robots try to transmit high-frame-rate video feeds.

platform	*sensing*	*locomotion*	*communication*
AIBO ERS-7 (`www.sonyaibo.net`)	camera	legged	wireless
Create (`www.irobot.com`) (with external laser device mounted on top)	laser	wheeled	wireless
"Fribbler" (= Scribbler: `www.parallax.com` + Fluke: `www.roboteducation.org`)	camera	wheeled	bluetooth
Mindstorms NXT (`mindstorms.lego.com`)	sonar	wheeled	bluetooth
SRV-1/ARM (`www.surveyor.com`)	camera	tracked	radio/wireless

Table 1. Robot platform capabilities

The human team member—the *operator*—is positioned physically away from the test arena so that her only view of the space is via camera images sent to her by the robots. The operator' interface is shown in Figure 4. The right half of the window shows a bird's eye view that indicates the position of each robot in the arena. The system uses vision-based localization (albeit somewhat unreliable due to the landmark identification problems mentioned above) and a particle filter to estimate the (x, y) location and orientation of each robot in the arena. The origin $(0, 0)$ of the robot's environment is defined as the middle of the arena (in the middle of the hallway), with positive x moving north and positive y moving east. Orientation (θ) is measured in degrees, with $0°$ facing east, $90°$ facing north, $180°$ facing west and $270°$ facing east. Returning to the operator interface in Figure 4, the upper left region contains a "robot's eye view" of the environment, from the perspective of one robot selected by the operator. The lower left region contains manual controls that the human can use to drive one robot at a time. Depending on the experimental conditions, the other robots are either idle when the human operator is not driving them (primarily this mode is used for taking experimental control measurements), or they are operating autonomously (most of the time).

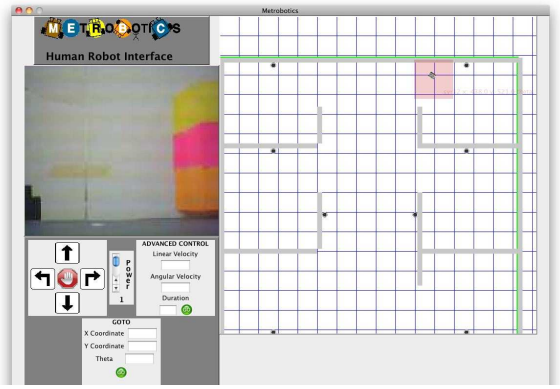


Fig. 4. Operator interface

4 Software Framework

Our software system employs a multi-layer architecture that combines multiple clients and multiple types of servers. A high-level overview of the system is shown in Figure 5. In the *agent layer*, the Central Server acts as the communication hub for all the components in the system, and is discussed separately, below. The Intelligence Engine supports system learning, task allocation and multi-robot coordination, as well as collaborative decision making with the human operator.

This component is not discussed in detail here; for further description, see [90]. The Database Manager logs system activity. It collects experimental data and maintains a database of known objects and other shared data structures (e.g., a map). The Object Recognizer identifies objects in the environment, by using the Open Source Computer Vision Library (OpenCV) [72] to perform feature extraction on robot imagery. Colored "blobs" are segmented and Canny edge detection [13] is applied to outline object shapes. A Naïve Bayes classifier [65] matches input images with previously tagged images from our database. The Operator Interface comprises the *human layer*, and was described in the previous section. The *robot layer* is detailed below (Section 4.1). Then, Section 4.2 discusses the overall system architecture and focuses on multi-server/multi-client aspects.

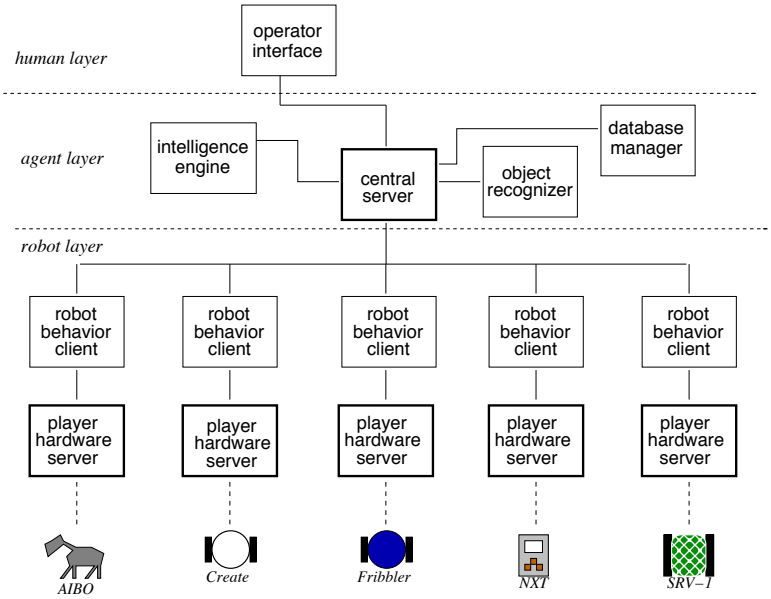


Fig. 5. The HRTeam system architecture. Each box is a process. The boxes outlined with thick borders are servers; the remaining boxes are clients.

4.1 Robot Layer

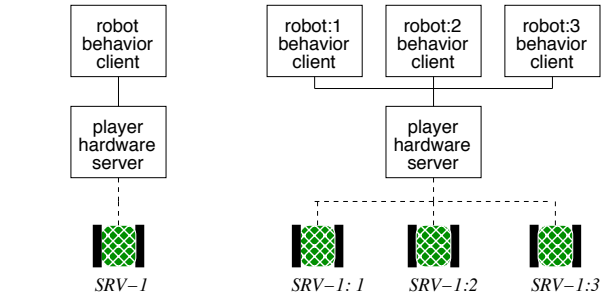
The robot layer is built on *Player/Stage*[2] [35, 97], a popular robot control and simulation environment. Player/Stage provides an open-source, modular client/-server framework for robotics programming that allows for unified control of

[2] `http://playerstage.sourceforge.net/`

multiple robot platforms. An abstract client class contains high-level robot control functionalities or behaviors (e.g., wall-following) and is extended to support the needs of a particular application. Hardware-specific drivers, implemented as servers, contain low-level sensor and actuator control functions (e.g., move forward or capture an image). In our framework, the client implements robot behaviors, such as perception, including some image processing, and low-level decision making for each robot. A platform-specific server, or driver, communicates directly with the robot hardware.

The advantage of Player/Stage is that, for each hardware platform we introduce onto our team, we need to write only one driver, and for each set of robot behaviors, we only need to write one behavior client, no matter how many different types of robot we want to run that behavior. We have adapted Player drivers for each of the five different robot platforms listed in Table 1. We have written one behavior client program that can control each of the robots in our system. A different behavior client process is instantiated for each robot, as explained below.

The use of Player/Stage presents an interesting system architecture question. It is possible to implement a system having a one-to-one correspondence between the robot behavior module, the hardware driver, and the physical robot (see Figure 6a). There may also be a one-to-many correspondence between the hardware driver and multiple physical robots (Figure 6b). In order to maintain the individuality of our robot team members, we always employ a one-to-one correspondence between robot behavior modules and physical robots.



(a) one robot, one hardware abstraction, one agent controller

(b) three robots one hardware abstraction, three agent controllers

Fig. 6. Player Framework

4.2 Central Server

An unusual aspect of our architecture is that there are multiple servers: the Central Server plus one Player hardware server for each (class of) robot platform.

The Central Server must be started up first, because it handles message passing and bookkeeping for the whole system. The Central Server keeps track of the instantiated components and of the robots that are connected to the system at any given time. All inter-process communication is handled asynchronously. All components have their own state machines; an example for the robot behavior client is shown in Figure 7. The components are designed to handle unexpected messages, as well as normal operations. The Central Server is written in C++ and establishes a server socket that binds to a particular host name and port number, establishing a point of communication for the entire system; then it listens for clients to connect. All of the processes in the system are multi-threaded, in order to handle communication asynchronously, independent of the process's primary functionality. For example, the Central Server creates a thread for each new client that connects to it, to allow asynchronous processing of messages between the Central Server and each client. Table 2 contains sample messages passed between the Central Server (CS) and a robot behavior client (RB). Table 3 contains sample messages that are passed between the Central Server (CS) and the operator interface (OI).

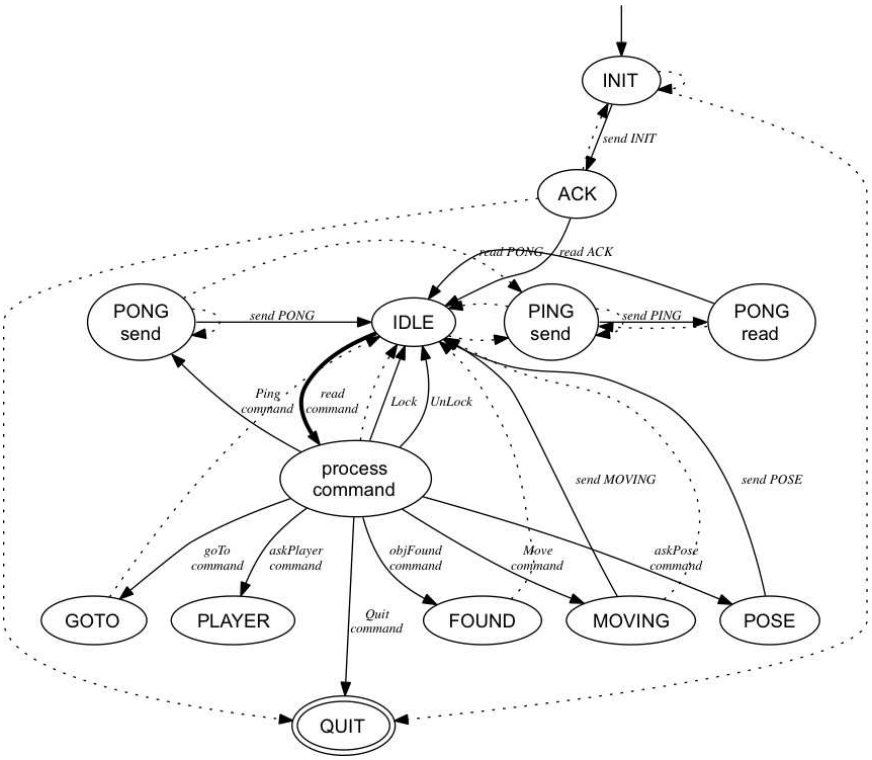


Fig. 7. State diagram for Robot Behavior client

RB $\rightarrow$ CS: **init** $\langle uds \rangle$ An RB sends this command to CS when it first logs in. $\langle uds \rangle$ stands for "unified data structure" that contains a string that identifies the type of robot (e.g., "aibo"); a string containing the name of the robot (e.g., "rosie"); a unique numeric identifier, which is treated like a session_id in the system and is determined by the server when a client first connects; and a list of the services that this robot provides, such as: "position2d", "camera", "distance", "contact".
RB $\leftarrow$ CS: **ack** $\langle id \rangle$ Upon receiving the **ack** command, the RB will set the value of the *id* field in its local copy of the unified data structure. $\langle id \rangle$ is a unique identifier (integer) that that CS sends to the RB to acknowledge its registration. It returns a unique ID number that the robot will need to use for all further communication, to identify itself in the system. This value is treated like a session_id.
RB $\leftarrow$ CS: **askpose** The CS sends an "askpose" message to the RB requesting information about its pose (location and heading).
RB $\rightarrow$ CS: **pose** $\langle x \rangle$ $\langle y \rangle$ $\langle \theta \rangle [\langle \rho \rangle]$ The RB sends back its (x, y) location and θ heading (degrees) within its environment. The last argument is confidence value, $0 \leq \rho \leq 1$, indicating the RB 's confidence in its location.
RB $\rightarrow$ CS: **broadcast found** $\langle color \rangle$ The RB sends this message whenever it finds an object of interest. CS strips "broadcast" part of the message and passes "found" $\langle color \rangle$ message to all connected clients, both robots and the OI.
RB $\leftarrow$ CS:**move** $\langle id \rangle$ $\langle x_velocity \rangle$ $\langle y_velocity \rangle$ $\langle angular_velocity \rangle$ The CS sends a "move" message to the robot requesting it to set its x, y and angular speeds to $\langle x_velocity \rangle, \langle y_velocity \rangle$ and $\langle angular_velocity \rangle$. If the $\langle id \rangle$ of the message does not match robots own id, the message is disregarded.
RB $\leftarrow$ CS: **goto** $\langle id \rangle$ $\langle map_x \rangle$ $\langle map_y \rangle$ The CS sends a "goto" message to the robot requesting it to move to a particular location, $(\langle map_x \rangle, \langle map_y \rangle)$, on the field. If the $\langle id \rangle$ of the message does not match robot's own id, the message is disregarded.
RB $\rightarrow$ CS: **moving** The RB sends back an acknowledgment that it has received the "move" command and is executing the command. OI does not need this confirmation, it will be used for data logging.

Table 2. Sample commands that flow between Central Server (CS) and Robot Behavior client (RB).

OI $\rightarrow$ CS **init** *uds* where *uds* is defined as in Table 2.
OI $\leftarrow$ CS **ack** $\langle id \rangle$ where $\langle id \rangle$ is defined as in Table 2.
OI $\rightarrow$ CS **askpose** $\langle id \rangle$ The OI sends "askpose" to CS to retreive the (x, y) location and θ heading of a particular robot, by attaching its $\langle id \rangle$. To retrieve pose information for all robots, $\langle id \rangle$ is set to -1.
OI $\leftarrow$ CS **pose** $\langle num_robots \rangle [\langle robot_pose_info \rangle]$ The CS sends back the number of robot pose information the message contains. Each pose information consists of robot's id, (x, y) location, θ heading (degrees) and confidence value, $0 \leq \rho \leq 1$, indicating the OI's confidence in its location.
OI $\rightarrow$ CS **askplayer** $\langle id \rangle$ The OI requests for player CS information that a particular robot is using. This information is needed to communicate directly with player CS to receive camera feed of the robot.
OI $\leftarrow$ CS **player** $\langle id \rangle$ $\langle player_ip \rangle$ $\langle player_port \rangle$ The CS sends back the player server information, $\langle player_ip \rangle, \langle player_port \rangle$ of the robot with id=$\langle id \rangle$.
OI $\leftarrow$ CS **found** The OI receives this message from the CS when a robot finds the object that the team is searching for. Currently it is used to stop the clock for the experiment.
OI $\rightarrow$ CS **move** $\langle id \rangle$ $\langle x_velocity \rangle$ $\langle y_velocity \rangle$ $\langle angular_velocity \rangle$ The OI sends a "move" message to the CS to pass it to robot with id=$\langle id \rangle$, requesting it to set its x, y and angular speeds to $\langle x_velocity \rangle, \langle y_velocity \rangle$ and $\langle angular_velocity \rangle$.
OI $\rightarrow$ CS **goto** $\langle id \rangle$ $\langle x \rangle$ $\langle y \rangle$ The OI sends a "goto" message to the CS to pass it to robot with id=$\langle id \rangle$, requesting it to move to a particular location, (x, y), on the field.
OI $\rightarrow$ CS **lock** $\langle id \rangle$ The OI sends a "lock" message to the OI, requesting to take control of the robot with id=$\langle id \rangle$.
OI $\rightarrow$ CS **unlock** $\langle id \rangle$ The OI sends an "unlock" message to the OI, requesting to release control of the robot with id=$\langle id \rangle$.

Table 3. Sample commands that flow between Central Server (CS) and Operator Interface (OI).

5 Lessons Learned

In this section, we describe some of the main lessons that we have learned from our work so far, largely in the form of problems we have had to contend with.

The main problem that we have faced has been getting the robots to localize while engaged in their exploration tasks. As mentioned above, we are using vision-based localization. The underlying approach is a standard particle filter, and the particular implementation we are using is one we developed for our Aibo-based RoboCup soccer team [74]. The main difference, as far as vision is concerned, between the Aibo, the Surveyor and the Fribbler—the robots that we have been using most often in our experiments—is that the last two have fixed cameras. It turns out that this has a large effect on their ability to see landmarks. When the robots start up, and move to maximize the number of landmarks they see, they localize relatively quickly. When they are carrying out their assigned task, however, which typically involves navigation through the test arena to explore a designated room, they often go for several minutes without seeing more than a single landmark clearly enough to recognize it. As a result, they rapidly become unsure of their location and have to spend time specifically relocalizing. This is in contrast to the Aibo, which can track its position quite effectively even with far fewer landmarks in the environment.

A subsidiary problem has been the wireless control of the robots. Several of our robots do not have sufficient on-board processing to run a controller (as mentioned in the previous section). Rather, they are controlled over a wireless connection, either 802.11, radio or Bluetooth. The first issue with wireless was mentioned above: 802.11 and radio interfere, and so if we are using the two modes of communication, we have to keep the robots physically separate. This, of course, adds another layer of complexity to control of the team. However, even if all the robots on the team use 802.11, there can still be issues. Even in the lab, where we have excellent wireless coverage, and little interference from other networks, it is easy to overload the bandwidth. With off-board processing, it is tempting to pull video off the robots at full-speed, but with more than two or three robots, this amount of traffic floods our local network. As a result we throttle the video feeds, though this naturally limits the use that both the robots and the human operator can make of the feeds. On the robot side, of course, this only makes the localization problem worse.

Finally, a more positive note. Despite the problems noted above, which could be eliminated if we used robots with multiple camera angles and more on-board processing[3], we have found our experience of using sub-$1000 robots to be a positive one. With the Player drivers we have developed, it is possible to use such robots for serious research purposes, and their cost means that with even a modest budget, it is possible to deploy a fleet of robots.

[3] Our future work will explore building cheap custom robots with Gumstix or Arduino controllers and multiple cameras to explore this option.

6 Summary

We have described the design and on-going implementation of our HRTeam framework, which we have developed to support studies in human/robot teamwork. Our philosophy has been to deploy multiple low-cost, limited-function robots, to force the necessity of collaboration in order to complete tasks. Our rough-and-ready laboratory environment offers special challenges, ranging from lighting variations and network interference to managing a suite of software components to control a heterogenous collection of hardware platforms. Several research activities are underway using the HRTeam framework. First, we are investigating ways to coordinate activity and allocate tasks to team members in a real-time, dynamic environment, concentrating on market-based mechanisms. Second, we are examining ways to incorporate real-time, dynamic input from the human operator into the multi-robot system. Finally, we are developing a participatory human/machine learning process to obtain reliability measures for the imaging data used in the localization process.

Acknowledgments

This work was supported by the National Science Foundation under #CNS-0851901 and #CNS-0520989, and by CUNY Collaborative Incentive Research Grant #1642.

References

1. Adams, J.A., Rani, P., Sarkar, N.: Mixed initiative interaction and robotic sys. In: Wkshp on Supervisory Control of Learning and Adaptive Sys, Tech Rept WS-04-10 (2004)
2. Alami, R., Robert, F., Ingrand, F., Suzuki, S.: Multi-robot cooperation through incremental plan-merging. In: Proc of the IEEE Conference on Robotics and Automation (1995)
3. Andrade-Cetto, J., Vidal-Calleja, T., Sanfeliu, A.: Multirobot C-SLAM: Simultaneous localization, control and mapping. In: Proc of the ICRA Workshop on Network Robot Systems (2005)
4. Atay, N., Bayazit, B.: Emergent task allocation for mobile robots. In: Proc of Robotics: Science and Systems Conference (2007)
5. Barlow, G., Henderson, T., Nelson, A., Grant, E.: Dynamic leadership protocol for S-Nets. In: Proc of the IEEE Intl Conference on Robotics and Automation (2004)
6. Bhattacharya, S., Candido, S., Hutchinson, S.: Motion strategies for surveillance. In: Proc of Robotics: Science and Systems Conference (2007)
7. Bhattacharya, S., Hutchinson, S.: Approximation schemes for two-player pursuit evasion games with visibility constraints. In: Proc of Robotics: Science and Systems Conference (2008)
8. Blank, D., Kumar, D., Meeden, L., Yanco, H.: Pyro: A python-based versatile programming environment for teaching robotics. ACM Journal on Educational Resources in Computing (JERIC) (2005)

9. Breazeal, C.: Toward sociable robotss. Robotics and Autonomous Systems 42 (2003)
10. Breazeal, C., Scassellati, B.: Robots that imitate humans. TRENDS in Cognitive Sciences 6(11) (2002)
11. Brumitt, B.L., Stentz, A.: Dynamic mission planning for multiple mobile robots. In: Proc of the IEEE Intl Conference on Robotics and Automation (1996)
12. Burke, J.L., Murphy, R.R.: Human-robot interaction in usar technical search: Two heads are better than one. In: Intl Workshop on Robot and Human Interactive Comm (2004)
13. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (1986)
14. Cao, Y.U., Fukunaga, A.S., Kahng, A.B.: Cooperative mobile robotics: antecedents and directions. Autonomous Robots 4(1) (1997)
15. Carbonell, J.R.: Mixed-initiative man-computer instructional dialogues. Tech. Rep. 1971, Bolt Beranek and Newman, Inc (1971)
16. Chaimowicz, L., Kumar, V., Campos, F.: A paradigm for dynamic coordination of multiple robots. Autonomous Robots 17(1) (2004)
17. Crasar: access oct 20. `crasar.org` (2010)
18. Dautenhahn, K.: A Paradigm Shift in Artificial Intelligence: Why Social Intelligence Matters in the Design and Development of Robots with Human-Like Intelligence. 50 Years of AI Journal LNCS 4850 (2007)
19. Dedeoglu, G., Sukhatme, G.S.: Landmark-based matching algorithm for cooperative mapping by autonomous robots. In: Distributed Autonomous Robotic Systems 4. Springer-Verlag (2000)
20. Dias, M.B., Zlot, R., Kalra, N., Stentz, A.: Market-based multirobot coordination: A survey and analysis. Tech. Rep. CMU-RI-TR-05-13, Carnegie Mellon University (2005)
21. Dias, M.B., Zlot, R., Zinck, M., Gonzalez, J.P., Stentz, A.: A versatile implementation of the traderbots approach for multirobot coordination. In: Proc of Intelligent Automated Systems (2004)
22. Dias, M.B., Zlot, R., Zinck, M., Stentz, A.: Robust multirobot coordination in dynamic environments. In: Proc of the IEEE Intl Conference on Robotics and Automation (2004)
23. Fenwick, J.W., Newman, P.M., Leonard, J.J.: Cooperative concurrent mapping and localization. In: Proc of the IEEE Conference on Robotics and Automation (2002)
24. Fink, J., Michael, N., Kumar, V.: Composition of vector fields for multi-robot manipulation via caging. In: Proc of Robotics: Science and Systems Conference (2007)
25. Finzi, A., Orlandini, A.: Human-Robot Interaction Through Mixed-Initiative Planning for Rescue and Search Rovers. Advances in Artificial Intelligence (AIIA) LNCS 3673 (2005)
26. Fong, T., Nourbakhsh, I., Dautenhahn, K.: A survey of socially interactive robots. Robotics and Autonomous Systems 42 (2003)
27. Fong, T., Thorpe, C., Baur, C.: Multi-Robot Remote Driving With Collaborative Control. IEEE Transactions on Industrial Electronics 50(4) (2003)
28. Fox, D., Burgard, W., Kruppa, H., Thrun, S.: A probabilistic approach to collaborative multi-robot localization. Autonomous Robots 8(3) (2000)
29. Freese, M., Matsuzawa, T., Oishi, Y., Debenest, P., Takita, K., Fukushima, E.F., Hirose, S.: Robotics-assisted demining with gryphon. Advanced Robotics 21(15) (2007)

30. Frias-Martinez, V., Sklar, E.I.: A framework for exploring role assignment in real-time, multiagent teams. In: The second European Workshop on Multi-Agent Systems (EUMAS) (2004)
31. Frias-Martinez, V., Sklar, E.I., Parsons, S.: Exploring auction mechanisms for role assignment in teams of autonomous robots. In: Proc of the Eighth RoboCup Intl Symposium (2004)
32. Gerkey, B.P., Mataríc, M.J.: Sold!: Auction methods for multi-robot control. IEEE Transactions on Robotics and Automation Special Issue on Multi-Robot Systems 18(5) (2002)
33. Gerkey, B.P., Mataríc, M.J.: Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In: Proc of the IEEE Intl Conference on Robotics and Automation (2003)
34. Gerkey, B.P., Mataríc, M.J.: A formal analysis and taxonomy c of task allocation in multi-robot systems. Intl Journal of Robotics Research 23(9) (2004)
35. Gerkey, B., Vaughan, R.T., Howard, A.: The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. In: Proc of the 11th Intl Conference on Advanced Robotics (ICAR) (2003)
36. Goodrich, M.A., Olsen, D.R., Crandall, J.W., Palmer, T.J.: Experiments in Adjustable Autonomy. In: IJCAI Workshop on Autonomy, Delegation, and Control: Interaction with Autonomous Agents (2001)
37. Goodrich, M.A., Schultz, A.C.: Human-robot interaction: a survey. Foundations and Trends in Human-Computer Interaction 1(3) (2007)
38. Habib, M.K.: Humanitarian Demining: Reality and the Challenge of Technology. Intl Journal of Advanced Robotic Systems 4(2) (2007)
39. Hajjdiab, H., Laganiere, R.: Vision-based multi-robot simultaneous localization and mapping. In: Procedings of the Canadian Conference on Computer and Robot Vision (2004)
40. Halász, A., Hsieh, M.A., Berman, S., Kumar, V.: Dynamic redistribution of a swarm of robots among multiple sites. In: IEEE/RSJ Intl Conference on Intelligent Robots and Systems (2005)
41. Hollinger, G., Singh, S.: Proofs and experiments in scalable, near-optimal search by multiple robots. In: Proc of Robotics: Science and Systems Conference (2008)
42. Hollinger, G., Singh, S., Djugash, J., Kehagias, A.: Efficient multi-robot search for a moving target. Intl Journal of Robotics Research 28(2) (2009)
43. Hollinger, G., Singh, S., Kehagias, A.: Efficient, guaranteed search with multi-agent teams. In: Proc of Robotics: Science and Systems Conference (2009)
44. Hong, J.H., Song, Y.S., Cho, S.B.: Mixed-initiative human-robot interaction using hierarchical bayesian networks. IEEE Transactions on Systems, Man and Cybernetics, Part A 37(6) (2007)
45. Horvitz, E.: Principles of mixed-initiative user interfaces. In: Proc of the Computer-Human Interaction Conference (CHI) (1999)
46. Howard, A.: Multi-robot simultaneous localization and mapping using particle filters. Journal of Robotics Research 25(12) (2006)
47. Huang, G.P., Trawny, N., Mourikis, A.I., Roumeliotis, S.I.: On the consistency of multi-robot cooperative localization. In: Proc of Robotics: Science and Systems Conference (2009)
48. Jacoff, A., Messina, E., Evans, J.: A standard test course for urban search and rescue robots. In: Proc of the Performance Metrics for Intelligent Systems Workshop (PerMIS) (2000)

49. Kaber, D.B., Wright, M.C., Sheik-Nainar, M.A.: Multimodal interface design for adaptive automation of a human-robot system. Intl Journal of Human-Computer Studies 64 (2006)
50. Kalra, N.: A market-based framework for tightly-coupled planned coordination in multirobot teams. Ph.D. thesis, The Robotics Institute, Carnegie Mellon University (2007)
51. Kalra, N., Ferguson, D., Stentz, A.: Hoplites: A market-based framework for complex tight coordination in multi-robot teams. In: Proc of the IEEE Intl Conference on Robotics and Automation (2005)
52. Kanda, T., Hirano, T., Eaton, D.: Interactive robots as social partners and peer tutors for children: A field trial. Human-Computer Interaction 19 (2004)
53. Kang, S., Lee, W., Kim, M., Shin, K.: ROBHAZ-rescue: rough-terrain negotiable teleoperated mobile robot for rescue mission. In: IEEE Intl Workshop on Safety, Security and Rescue Robotics (2005)
54. Koenig, S., Keskinocak, P., Tovey, C.: Progress on agent coordination with cooperative auctions. In: Proc of the AAAI Conference on Artificial Intelligence (2010)
55. Lagoudakis, M., Berhault, M., Koenig, S., Keskinocak, P., Kelywegt, A.: Simple auctions with performance guarantees for multi-robot task allocation. In: Proc of Int'l Conference on Intelligent Robotics and Systems (IROS) (2004)
56. Lagoudakis, M., Markakis, V., Kempe, D., Keskinocak, P., Koenig, S., Kleywegt, A., Tovey, C., Meyerson, A., Jain, S.: Auction-based multi-robot routing. In: Proc of Robotics: Science and Systems Conference (2005)
57. LaValle, S.M., Hutchinson, S.A.: Optimal motion planning for multiple robots having independent goals. IEEE Transactions on Robotics and Automation 14(6) (1998)
58. Lázaro, M.T., Castellanos, J.A.: Localization of probabilistic robot formations in SLAM. In: Proc of the IEEE Intl Conference on Robotics and Automation (2010)
59. Lerman, K., Galstyan, A.: Mathematical model of foraging in a group of robots: Effect of interference. Autonomous Robots 13(2) (2002)
60. Lerman, K., Jones, C.V., Galstyan, A., Mataríc, M.J.: Analysis of dynamic task allocation in multi-robot systems. Intl Journal of Robotics Research 25(3) (2006)
61. Liu, W., Winfield, A.F.T., Sa, J., Chen, J., Dou, L.: Towards energy optimization: Emergent task allocation in a swarm of foraging robots. Adaptive Behavior 15(3) (2007)
62. Mataric, M., Sukhatme, G., Ostergaard, E.: Multi-robot task allocation in uncertain environments. Autonomous Robots (2003)
63. McLurkin, J., Yamins, D.: Dynamic task assignment in robot swarms. In: Proc of Robotics: Science and Systems Conference (2005)
64. Michael, N., Zavlanos, M.M., Kumar, V., Pappas, G.J.: Distributed multi-robot task assignment and formation control. In: Proc of the IEEE Intl Conference on Robotics and Automation (2008)
65. Mitchell, T.M.: Machine Learning. McGraw Hill (2005)
66. Murphy, R.R.: Marsupial and shape-shifting robots for urban search and rescue. IEEE Intelligent Systems 15(2) (2000)
67. Murphy, R.R., Casper, J., Micire, M.: Potential tasks and research issues for mobile robots in robocup rescue. In: Robot Soccer World Cup IV, LNAI 2019. Springer Verlag (2001)
68. Murrieta-Cid, R., Muppirala, T., Sarmiento, A., Bhattacharya, S., Hutchinson, S.: Surveillance strategies for a pursuer with finite sensor range. Intl Journal of Robotics Research (2007)

69. Nelson, A., Grant, E., Barlow, G., Henderson, T.: A colony of robots using vision sensing and evolved neural controllers. In: IEEE/RSJ Intl Conference on Intelligent Robots and Systems (2003)
70. Nelson, A., Grant, E., Henderson, T.: Evolution of neural controllers for competitive game playing with teams of mobile robots. Robotics and Autonomous Systems 46 (2004)
71. Nerurkar, E.D., Roumeliotis, S.I., Martinelli, A.: Distributed maximum a posteriori estimation for multi-robot cooperative localization. In: Proc of the IEEE Intl Conference on Robotics and Automation (2009)
72. Open Source Computer Vision Library (OpenCV). `http://sourceforge.net/projects/opencvlibrary/`
73. Ota, J.: Multi-agent robot systems as distributed autonomous systems. Advanced Engineering Informatics 20 (2006)
74. Ozgelen, A.T., Kammet, J., Marcinkiewicz, M., Parsons, S., Sklar, E.I.: The 2007 MetroBots Four-legged League Team Description Paper. In: RoboCup 2007: Robot Soccer World Cup XI (2007)
75. Parker, L.E.: Cooperative robotics for multi-target observation. Intelligent Automation and Soft Computing 5(1) (1999)
76. Pereira, G.A.S., Campos, M.F.M., Kumar, V.: Decentralized algorithms for multi-robot manipulation via caging. Intl Journal of Robotics Research (2004)
77. Rooker, M.N., Birk, A.: Multi-robot exploration under the constraints of wireless networking. Control Engineering Practice 15 (2007)
78. Roomba: access oct 20. `www.irobot.com` (2010)
79. Rooy, D., Ritter, F., St Amant, R.: Using a simulated user to explore human-robot interfaces. In: ACT-R Workshop (2002)
80. Royset, J., Sato, H.: Route optimization for multiple searchers. Tech. rep., Naval Postgraduate School, Monterey, CA (2009), `http://faculty.nps.edu/joroyset/docs/RoysetSato_MultiSearcher.pdf`
81. Sandini, G., Metta, G., Vernon, D.: The iCub Cognitive Humanoid Robot: An Open-System Research Platform for Enactive Cognition. 50 Years of AI Journal LNCS 4850 (2007)
82. Santana, P.F., Barata, J., Correia, L.: Sustainable Robots for Humanitarian Demining. Intl Journal of Advanced Robotic Systems 4(2) (2007)
83. Sariel, S., Balch, T.: Efficient bids on task allocation for multi-robot exploration. In: Proc of the Nineteenth Intl Florida Artificial Intelligence Research Society Conference (2006)
84. Sarmiento, A., Murrieta-Cid, R., Hutchinson, S.: A multi-robot strategy for rapidly searching a polygonal environment. In: Proc of the 9th Ibero-American Conference on Artificial Intelligence (2004)
85. Scerri, P., Pynadath, D.V., Tambe, M.: Why the elf acted autonomously: Towards a theory of adjustable autonomy. In: Proc of the Intl Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS) (2002)
86. Severinson-Eklundh, K., Green, A., Hüttenrauch, H.: Social and collaborative aspects of interaction with a service robot. Tech. Rep. IPLab-208, Royal Institute of Technology, Stockholm (January 2003)
87. Shah, K., Meng, Y.: Communication-efficient dynamic task scheduling for heterogeneous multi-robot systems. In: Proc of the IEEE Intl Symposium on Computational Intelligence in Robotics and Automation (2007)
88. de Silva, V., Ghrist, R., Muhammad, A.: Blind swarms for coverage in 2-d. In: Proc of Robotics: Science and Systems Conference (2005)

89. Simmons, R., Apfelbaum, D., Burgard, W., Fox, D., Moor, M., Thrun, S., Younes, H.: Coordination for multi-robot exploration and mapping. In: Proc of the 17th National Conference on Artificial Intelligence (2000)
90. Sklar, E.I., Epstein, S.L., Parsons, S., Ozgelen, A.T., Munoz, J.P.: A framework in which robots and humans help each other. In: Proc of the AAAI Symposium Help Me Help You: Bridging the Gaps in Human-Agent Collaboration (2011)
91. Spletzer, J., Das, A.K., Fierro, R., Taylor, C.J., Kumar, V., Ostrowski, J.P.: Cooperative localization and control for multi-robot manipulation. In: Proc of the IEEE/RSJ Intl Conference on Intelligent Robots (2001)
92. Stone, P., Veloso, M.: Communication in domains with unreliable, single-channel, low-bandwidth communication. In: Proc of the Intl Joint Conference on Artificial Intelligence (IJCAI) (1998)
93. Svestka, P., Overmars, M.H.: Coordinated path planning for multiple robots. Robotics and Autonomous Systems 23 (1998)
94. Thrun, S., Liu, Y., Koller, D., Ng, A.Y., Ghahramani, Z., Durrant-Whyte, H.: Simultaneous localization and mapping with sparse extended information filters. Journal of Robotics Research (2004)
95. Tyrer, H., Alwan, M., Demiris, G., He, Z., Keller, J., Skubic, M., Rantz, M.: Technology for successful aging. In: Proc of Engineering in Medicine and Biology Society (2006)
96. Vail, D., Veloso, M.: Dynamic multi-robot coordination. In: Schultz, A.C., Parker, L.E., Schneider, F.E. (eds.) Multi-Robot Systems: From Swarms to Intelligent Automata. Kluwer (2003)
97. Vaughan, R.T., Gerkey, B.: Really Reusable Robot Code and the Player/Stage Project. In: Brugali, D. (ed.) Software Engineering for Experimental Robotics. Springer (2007)
98. Wegner, R., Anderson, J.: Agent-based support for balancing teleoperation and autonomy in urban search and rescue. Intl Journal of Robotics and Automation 21(2) (2006)
99. Williams, K., Burdick, J.: Multi-robot boundary coverage with plan revision. In: Proc of the IEEE Conference on Robotics and Automation (2006)
100. Woods, D., Tittle, J., Feil, M., Roesler, A.: Envisioning human-robot coordination in future operations. IEEE Transactions on Systems, Man and Cybernetics, Part C 34(2) (2004)
101. Yamashita, A., Arai, T., Ota, J., Asama, H.: motion planning of multiple mobile robots for cooperative manipulation and transportation. IEEE Transactions on Robotics and Automation 19(2) (2003)
102. Yanco, H., Baker, M., Casey, R., Keyes, B., Thoren, P., Drury, J.L., Few, D., Nielsen, C., Bruemmer, D.: Analysis of Human-Robot Interaction for Urban Search and Rescue. In: Proc of the IEEE Intl Workshop on Safety, Security and Rescue Robotics (2006)
103. Zheng, X., Koenig, S.: K-swaps: Cooperative negotiation for solving task-allocation problems. In: Proc of the Intl Joint Conference on Artificial Intelligence (2009)
104. Zlot, R., Stentz, A., Dias, M.B., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proc of the IEEE Conference on Robotics and Automation (2002)