

Robocup Rescue - Virtual Robots Team STEEL (USA) MrCS - The Multirobot Control System

Prasanna Velagapudi¹, Alexander Kleiner¹, Nathan Brooks¹, Paul Scerri¹,
Michael Lewis², and Katia Sycara¹

¹ Carnegie Mellon University, Pittsburgh PA, USA

² University of Pittsburgh, Pittsburgh PA, USA

Abstract. This paper describes the software system supporting the Carnegie Mellon/Univ. of Pittsburgh team of simulated search and rescue robots in the Robocup Rescue 2010 Virtual Robots competition. Building on the Machinetta agent software, robot command and control is decomposed into a hierarchy of subtasks managed by independent agents both on the robot and colocated with human operators. By encapsulating all robot and human operator interactions into interfaces to these agents, the system can perform with a high level of robustness and reusability. As in previous years, the entire code base is portable and platform-independent, running entirely in Java.

1 Introduction

In human-robot interaction, how and when the operator intervenes in the robotic system are the two predominant issues [Endsley, 1996]. How a human works with the system is a function of the level of autonomy (LOA), which describes the static function assignments between the human and the robot. The LOA can range from full manual control to full autonomy, with intermediate levels of LOA generally being superior to full autonomy or full manual control. This is because an LOA that is too high leads to degradation in manual or mental skill, loss of situation awareness, decision bias, and decrease in vigilance. The low LOA of full manual control leads to high mental demand, human decision bias, complacency, boredom, and inconsistent control behavior, all of which degrade performance. In systems with adaptive autonomy (AA), the allocation of control between the human and the robot can be dynamically changed and is usually triggered by a critical event, performance measurement, operator's workload, or the operator model. Carefully calibrated AA maximizes the amount of time the human operator can spend doing tasks which humans perform better than robots, such as victim identification and navigation of robots out of stuck or dangerous positions, problems present with the current state-of-the-art for robots. If utilized effectively, it is clearly the case that adding robots to a search and rescue team will improve the speed at which an area can be searched, leading to

more victims being saved. However, utilizing more robots for searching for victims with appropriate AA requires both effective autonomy and effective means for an operator to interact with the team. Thus, the focus of the 2010 Steel team is *user-centered autonomy*. The autonomy will be designed specifically with the operator in mind, performing routine tasks and carefully utilizing the operator’s time for the tasks they are most suited to. First we will look at the capabilities of our software packages.

2 Machinetta

The teamwork algorithms used in MrCS are general algorithms that have been shown to be effective in a range of domains [Tambe, 1997]. To take advantage of this generality, the emerging standard approach is to encapsulate the algorithms in a reusable software proxy. Each team member has a proxy that he works with closely, and the proxies work together to implement the teamwork. The current version of the proxies utilized in MrCS is Machinetta [Scerri *et al.*, 2005b], which is implemented in Java and is freely available on the internet. This type of proxy differs from many other multi-agent toolkits in that it provides the coordination algorithms, e.g., algorithms for allocating tasks, as opposed to the infrastructure, e.g., APIs for reliable communication.

The Machinetta software consists of five main modules, three of which are domain-independent and two of which are tailored for specific domains. The three domain-independent modules are designed for coordination reasoning, maintaining local beliefs (state), and adjustable autonomy. The domain-specific modules are designed for communication between proxies and communication between a proxy and a team member. These modules interact with each other only via the local state with a blackboard design and are designed to be “plug and play”. This means, for examples, that new adjustable autonomy algorithms can be used with existing coordination algorithms.

The coordination reasoning is responsible for reasoning about interactions with other proxies, thus implementing the coordination algorithms. The adjustable autonomy algorithms reason about the interaction with the team member, providing the possibility for the team member rather than the proxy to make any coordination decision. For example, the adjustable autonomy module can reason that a decision to accept the role of rescuer for a civilian in a burning building should be made by the human who will enter the building rather than the proxy. In practice, the overwhelming majority of coordination decisions are made by the proxies, and only key decisions are referred to the human operators. Teams of proxies implement team-oriented plans (TOPs) which describe joint activities to be performed in terms of the individual roles to be performed and any constraints on those roles. Typically, TOPs are instantiated dynamically from TOP templates at run-time when pre-conditions associated with the templates are filled. Constraints between these roles specify interactions, such as the required execution ordering and whether one role can be performed if another is not currently being performed. It is important to note that TOPs

do not specify the coordination or communication required to execute a plan. Instead, the proxy determines the coordination that should be performed.

Current versions of Machinetta include state-of-the-art algorithms for plan instantiation, role allocation, information sharing, task deconfliction, and adjustable autonomy. Many of these algorithms utilize a logical associates network that statically connects all team members. The associates network is a scale free network which allows the team to balance the complexity of needing to know about all the team and maintaining cohesion. The associates network’s key algorithms, including role allocation, resource allocation, information sharing, and plan instantiation, are based on the use of tokens that are “pushed” onto the network and routed to where they are required by the proxies. For example, the role allocation algorithm LA-DCOP [Scerri *et al.*, 2005a] represents each role to be allocated with a token and pushes the tokens onto the network until a sufficiently capable and available team member is found to execute the role. The implementation of the coordination algorithms uses the abstraction of a simple mobile agent to implement the tokens, leading to robust and efficient software.

3 MrCS

The system architecture of MrCS is shown in Figure 1. Each robot connects with Machinetta through a robot driver that controls the robot on both low and middle levels of control. For low-level control, it serves as a broker that translates robot sensory data into local beliefs and that translates the exploration plan into robot control commands (e.g., wheel speed control). For middle-level control, the driver analyzes robot sensory data to perceive its states and local environment. Then, based on this perception, the driver overrides the control commands when it is necessary to ensure safe movement. Possible adjustments include changing the direction of motion to avoid obstacles and recovering from becoming immobilized and from a dangerous pose. When the robot is in an idle state, laser data analysis allows the driver to generate potential exploration plans (e.g., the destination and the path to the destination). In addition, when the robot senses a potential victim³, the driver immediately stops the robot and generates a plan to inspect the potential victim. However, instead of executing the plans immediately, the driver sends them to the Machinetta proxy to trigger TOPs. With Machinetta’s role allocation algorithm, the robots and the human cooperate with each other to find the “best” robot to execute a plan. Here the “best” robot is defined as the robot that can find a route to the destination with the least cost, i.e., the shortest weighted travel length. The operator connects with Machinetta through the user interface agent. This agent collects the robot team’s beliefs and visually represents them on the interface. It also transfers the operator’s commands in the form of a Machinetta proxy’s beliefs and passes them to the proxies network to allow human intervention in the loop cooperation. The operator can intervene with the robot team on three levels. On the lowest level,

³ This functionality was added for RoboCup. In the competition, a faked “super” victim sensor was introduced to allow a robot automatically sense a potential victim.

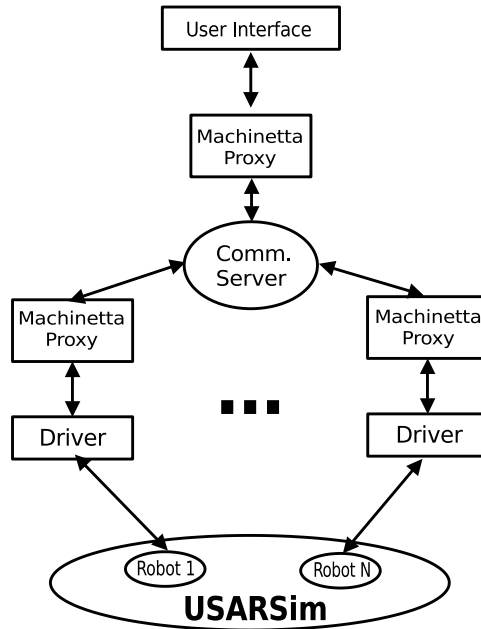


Fig. 1: MrCS architecture.

the operator takes over an individual robot's autonomy to teleoperate it. On the middle level, the operator interacts with a robot via editing its exploration plan. For example, the operator is allowed to delete a robot's plan to force it to stop and regenerate a plan or issue a new plan (a series of waypoints) to change its exploration behavior. On the highest level, the operator intervenes with the entire robot team via issuing priority areas. The priority will impact the cost calculation in role allocation and therefore affects the regions that the robots will explore.

In this human-robot team, the human maintains the highest authority to adjust the robot team's behavior. For example, the human can change a plan during plan execution, and this plan can be further adjusted by the robot to avoid obstacles or a dangerous pose. When critical events, such as sensing a potential victim or being in a dangerous pose, occur, the robot adjusts its own behavior and informs the operator. In this case, the robot initiates the interaction and the operator can either accept the robot's adjustment or change the robot's plan. One of the challenges in a mixed-initiative system is that the user may fail to maintain situation awareness of the robot team and of the individual robots when control switching and may therefore make faulty decisions. Moreover, as the team size increases, the interventions from the robots may overwhelm the operator's cognitive resources [McFarlane and Latorella, 2002] and the operator may be limited to reacting to the robots instead of proactively controlling the

robots [Trouvain, 2003]. We address these issues in the user interface design described below.

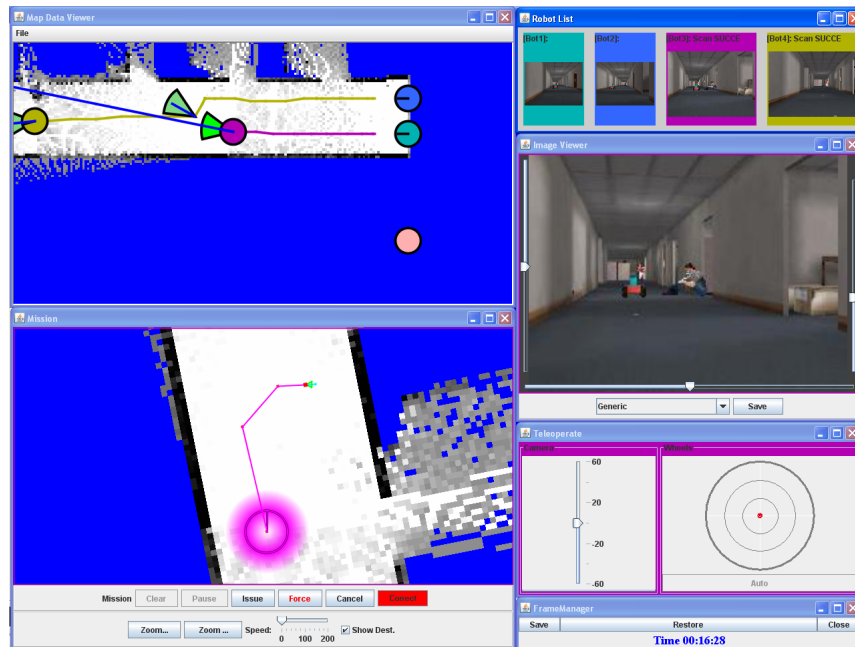


Fig. 2: MrCS user interface.

The user interface of MrCS is shown in Figure 2. The interface is reconfigurable to allow the user to resize the components or change the layout. Shown in the figure is a configuration that we used in the RoboCup 2006 competition in which a single operator controls six robots. On the upper and center portions of the left-hand side are the robot list and team map panels, which show the operator an overview of the team. The destination of each of robot is displayed on the map to help the user perceive team performance. Using this display, the operator is also able to control regional priorities by drawing rectangles on the map. On the center and lower portions of the right-hand side are the camera view and mission control panels, which allow the operator to maintain situation awareness of an individual robot and to edit its exploration plan. On the mission panel, the map and all nearby robots and their destinations are represented to provide partial team awareness so that the operator can switch between contexts while moving control from one robot to another. The lower portion of the right-hand side is a teleoperation panel that allows the operator to teleoperate a robot. On the interface, interruptions from the robots are mitigated by using principles of etiquette in user interface design [Nass, 2004]. When the robot needs the operator's attention, such as when sensing a victim or being in dangerous pose, the

system will not display a pop-up window but will instead temporarily change the mission panel's size and background color or flash the robot's thumbnail picture in the robot list panel to inform the operator that a robot needs to be checked. This silent form of alert allows the operator to work at his own pace and respond to the robots when able.

We utilized MrCS in the RoboCup 2006 virtual robot competition. The four days of practice showed that, with mixed-initiative control and a simple cooperation algorithm (avoiding duplicate exploration effort), a single operator can control six robots. The overall performance during the competition was superior to other human-involved systems, and the final score was comparable to other full autonomy systems even after our score was divided by four due to the use of an operator [Balakirsky *et al.*, 2007].

4 Concept

The overall Steel team system can be divided into two parts. The first part is the autonomous robot behaviors that provide the infrastructure which the operator will utilize. The second part is the interface through which the operator monitors and interacts with the team. Below we describe some of the team elements of both these parts.

5 Autonomy

Previous Steel teams have utilized some minimal levels of autonomy, but in 2010 four new components will make the Steel robots capable of significant amounts of autonomous activity. The four components are: (1) vision-coverage based autonomous search; (2) user-directed high-level exploration; (3) self-reflection; and (4) detailed victim localization. Below we give a short overview of the planned capabilities of each of these components.

5.1 Vision-coverage based autonomous search

Finding victims eventually relies on the operator looking at an image with the victim visible. Thus, a search should attempt to provide images of the whole environment as quickly as possible. However, most autonomous search and autonomous SLAM algorithms focus on coverage with laser scanners to build maps. While the laser scan results and generated maps are critical input into determining whether cameras have seen the complete environment, their coverage is not what should be optimized. Technically, vision-coverage will work as effectively as traditional laser-based SLAM, but build the equivalent of occupancy grids based on what the robot believes is the quality of imagery that has been collected for each location.

One approach being considered is implementing a queue of view objects to minimize the context switching penalty incurred by frequently shifting between

the vantage points of several robots. The view objects consist of a camera image and the matching laser scan readings. When a robot is facing an area which has not been marked as being clear of victims, it adds the view object onto the queue. The laser scan reading is projected onto the 2D map and intersecting cells of the victim identification map are marked as "pending". The projected area is confined to be within a minimum and maximum laser scan distance to prevent errors due to narrow fields of view or insufficient camera resolution, respectively. The queue is prioritized based on the calculated information gain of the projected area of the laser scan. The human operator monitoring the queue views the images and uses a simple button interface to indicate whether or not there is a victim in the image or if a different view is needed. In the event that the operator indicates there is a victim, a marker is placed on the map automatically. If the operator decides no victims are present, the cells in the victim identification map covered by the projected laser scan area are marked as being clear. If the operator is unsure, a task to revisit the area is added to the task queue. With this approach, there is no cost for re-orientation of the operator to a recorded or live camera feed and the operator can quickly evaluate images while the robots move on to explore other areas.

5.2 User-directed exploration

Simultaneous Localization And Mapping (SLAM)

For our 2D representation of the environment, we generate maps from laser-based scan matching. During the last decades a rich set of solutions for building maps from 2D laser range data has been proposed, such as [Lu and Milios, 1997; Gutmann, 2000; Hähnel, 2005]. In contrast to scan matching methods, more sophisticated methods, such as *FastSlam* [Montemerlo *et al.*, 2002], and *GMapping* [Grisetti *et al.*, 2005], were introduced, that correct the entire map at once when loop-closures, i.e., re-visits of places, are detected.

Although existing methods are capable of dealing with sensor noise, they do require reasonable pose estimates, e.g., from wheel odometry, as an initial guess for the mapping system. As commonly known, wheel odometry tends to become unreliable given an unpredictable amount of wheel slip, which is frequently the case on the rough terrain encountered in USAR missions. Furthermore, methods performing loop-closures are mostly not applicable in real-time since their computational needs can unpredictably increase in unknown environments.

Exploration and Path Planning

The mapping approach utilized for our robot team focuses on the application scenario of realistic teleoperation. Under certain constraints, such as low visibility and rough terrain, first responder teleoperation leads to very noisy and unusual data. For example, due to environmental make-up and failures in control, laser scans are frequently taken under a varying roll and pitch angle, making it

difficult to reliably find correspondences from successive measurements. In contrast to artificially generated data logs, logs from teleoperation seldom contain loops. Most existing methods follow the principle of minimizing the squared sum of error distances between successive scans by searching over scan transformations, i.e., rotations and translations. Scan point correspondences are decided only once before the search starts based on the Euclidean distance. In contrast to other methods, our scan matching approach re-considers data associations during the search, which remarkably increases the robustness of scan matching on rough terrain. The algorithm processes data from the laser range finder and gyroscope only, making it independent from odometry failures, which are likely occur in such domains, e.g., due to slipping tracks. The mapping approach has been extensively tested on robot platforms designed for teleoperation in critical situations, such as bomb disposal. Furthermore, the system was evaluated in a test maze by first responders during the Disaster City event in Texas in 2008. Experiments conducted within different environments show that the system yields comparably accurate maps in real-time when compared to more sophisticated, offline methods, such as Rao-Blackwellized SLAM. More details on the utilized mapping approach are found in [Kleiner and Dornhege, 2009].

While these mapping algorithms are capable of being completely autonomous, they can be inefficient because they do not take advantage of environment features that are easily understood by an operator, especially if that operator has access to imagery from the robots. For example, if robots start at the end of a long corridor with offices all the way down, it will take a significant amount of time for a SLAM algorithm to work this out and the search may be quite inefficient. However, an operator seeing an image will immediately recognize the type of environment and may be able to design a much more efficient search strategy, for example, sending some robots to the end of the corridor to explore beyond the corridor, while a small number are left to check the offices. In the case of a cluttered environment, a robot will spend a great deal of time navigating around chairs and tables in order to fully map the wall behind the objects using its laser scanner. An operator can look at the sparsely populated 2D map and camera feed of the robot and possibly discern if there are victims present. If a victim is present, the operator can mark the map accordingly; if not, the operator can annotate the map by drawing a box around the sparsely, yet sufficiently, outlined walls of the room and mark the entire room as being clear, allowing the robot team to move on and explore other areas.

We will build the capability into the autonomous search for an operator to direct one robot or a group of robots to go directly to some location and continue the search from there. While the search will remain overwhelmingly autonomous, this ability to provide some high-level input is expected to make the search significantly more efficient, especially in cases where the operator accurately recognizes the type of environment and can utilize background knowledge to guide the search.

5.3 Self-reflection

In environments that are not simple and open, it is inevitable that robots will get stuck in positions from which they require human help to escape. If the team of robots is large, the operator can spend a large amount of their time simply checking whether the robots need any help. This distracts them from more important and useful tasks and can dramatically limit the number of robots a single operator can manage. We intend to reduce the time spent monitoring robots for problems by giving robots an ability to self-reflect and determine whether they are in a position that requires human help. We have a simple version of self-reflection implemented that looks at the pose of the robot, whether it is progressing along its planned path and whether it has been forced to re-plan many times in order to decide whether human input is required. We hope that this ability of self-reflection will eventually free the operator from all the time they currently spend monitoring the robots for problems.

5.4 Detailed victim localization

Another large drain on operator time is the detailed robot manipulation and cognitive processing required to accurately mark a victim on a map, once they believe they have seen a victim. The typical process would be to take over tele-operation of the robot and maneuver it around until it was clear there was or was not a victim and then attempt to match up what they are seeing with the map and accurately mark the victim. We are planning to develop an autonomous behavior that collects detailed information, with lots of different image angles in a specific area. The idea is that the operator could indicate that they think there is a victim in some location and then the robot would do all the time-consuming detailed maneuvering to get the right information for the operator to quickly confirm the victim's presence and mark their location. The autonomous behavior will use laser scan data to determine a range of angles that images should be taken from and then geo-reference those images into a coherent picture of the small part of the environment. We anticipate that this autonomous capability will free the operator from much of the time they currently spend tele-operating the robots.

5.5 User interface

The 2010 Steel user interface will build on the Multi-robot Control System (MrCS) interface that has been used in several competitions and been the basis for large human-factors experiments. This year, three major new capabilities are planned: (1) focusing user on new video; (2) prioritization of robots needing assistance; and (3) additional support for victim localization. Below we give a short overview of the planned capabilities of each of these components.

Identifying New Video

One of the remaining key tasks for the operator is to monitor video streams looking for victims. As the number of robots in the team is increased, this task becomes more time-consuming and difficult. However, some portion of the imagery shown to the operator will be a part of the environment already seen. For example, if a robot moves into and then out of an office, most of the time the robot spends coming out of the office will not provide new imagery or when multiple robots move down the same corridor. A convenient side-effect of the vision-based exploration is that the robots will be keeping track of when they are collecting imagery of a part of the environment that they have not yet shown the user. We will use this information to somehow highlight the video streams showing new images. We anticipate that this will dramatically lower the number of video streams the user will need to monitor at any time, reducing the danger of the operator missing victims and freeing up their time for other activities.

Prioritization of Robot Assistance

When there are many robots searching an environment, not all of them will be in equally useful positions. For example, midway through the search, some of the robots may have completely searched the part of the environment that they are in and have a long way to go to get to a new frontier. If there are multiple robots needing operator assistance, there may be different values to the overall objective for assisting each of the robots. We have been developing heuristics for assessing the relative importance of each robot to the team's medium term performance. This value will be used to prioritize assistance tasks for the operator, allowing them to focus on the most important tasks. In some cases, the operator may even decide that the robots requiring assistance are of low enough value to leave them stuck and concentrate on other tasks.

Victim Localization Aide

As noted above, significant time is used by the operator managing the robot to confirm victim sightings so we are exploring new autonomous behaviors to get detailed imagery in areas that the operator believes there might be a victim. From the operator's perspective, they will initiate the behavior simply by clicking on a location in a video stream. The click location will be mapped to some location on the ground by referencing to the laser scan of that area. Once the robot has done the data collection, it will be presented to the user, stitched together into a 3D representation of the world. When time permits, the user will be able to look at the 3D representation and click on the location of the victim (if any). This click will be mapped back to a 2D location where the victim can be marked on the map.

6 Conclusion

The MrCS system addresses the complex problem of urban search and rescue with a hierarchy of simple, robust solutions. Rather than solely depending on autonomy algorithms or operator skills, tasks are decomposed into cooperative efforts between robots and operators, allowing for speed and efficiency while minimizing the frequency and criticality of failures. For Robocup Rescue 2010 we have chosen to strengthen core robot features such as SLAM and path planning while adding new concepts to further maximize the use of the human operator's attention for tasks such as identifying victims and assisting stuck robots. Our updates to autonomy and user interface combine together to form a user-centered autonomy.

References

- [Balakirsky *et al.*, 2007] S. Balakirsky, S. Carpin, A. Kleiner, M. Lewis, A. Visser, J. Wang, and V.A. Ziparo. Towards heterogeneous robot teams for disaster mitigation: Results and performance metrics from robocup rescue. *Journal of Field Robotics*, 24(11–12):943–967, 2007.
- [Endsley, 1996] M.R. Endsley. Automation and situation awareness. *Automation and human performance: Theory and applications*, pages 163–181, 1996.
- [Grisetti *et al.*, 2005] G. Grisetti, Stachniss C., and Burgard W. Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective re-sampling. pages 667–672, Barcelona, Spain, 2005.
- [Gutmann, 2000] J.-S. Gutmann. *Robuste Navigation autonomer mobiler Systeme*. PhD thesis, Albert-Ludwigs-Universität at Freiburg, 2000. ISBN 3-89838-241-9.
- [Hähnel, 2005] D. Hähnel. *Mapping with Mobile Robots*. PhD thesis, Universität Freiburg, Freiburg, Deutschland, 2005.
- [Kleiner and Dornhege, 2009] A. Kleiner and C. Dornhege. Operator-assistive mapping in harsh environments. Denver, USA, 2009.
- [Lu and Milios, 1997] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2d range scans. *J. Intell. Robotics Syst.*, 18(3):249–275, 1997.
- [McFarlane and Latorella, 2002] D.C. McFarlane and K.A. Latorella. The scope and importance of human interruption in human-computer interaction design. *Human-Computer Interaction*, 17(1):1–61, 2002.
- [Montemerlo *et al.*, 2002] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fast-SLAM: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.
- [Nass, 2004] C. Nass. Etiquette equality: exhibitions and expectations of computer politeness. *Communications of the ACM*, 47(4):35–37, 2004.
- [Scerri *et al.*, 2005a] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Allocating tasks in extreme teams. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, page 734. ACM, 2005.
- [Scerri *et al.*, 2005b] P. Scerri, E. Liao, J. Lai, K. Sycara, Y. Xu, and M. Lewis. Coordinating very large groups of wide area search munitions. *Theory and Algorithms for Cooperative Systems*, 2005.

- [Tambe, 1997] M. Tambe. Towards Flexible Teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.
- [Trouvain, 2003] Schlick C. Mevert M. Trouvain, B. Comparison of a map- vs. camera-based user interface in a multi-robot navigation task. In *Proceedings of the 2003 International Conference on Robotics and Automation*, pages 3224–3231, 2003.