MARKET-BASED MULTI-ROBOT PLANNING IN A DISTRIBUTED LAYERED ARCHITECTURE

Dani Goldberg, Vincent Cicirello, M. Bernardine Dias
Reid Simmons, Stephen Smith, Anthony Stentz
Carnegie Mellon University, Robotics Institute
5000 Forbes Ave, Pittsburgh, PA 15213
(danig,cicirello,mbdias,reids,sfs,axs)@cs.cmu.edu

Abstract

This paper presents a market-based, multi-robot planning capability, designed as part of a distributed, layered architecture for multi-robot control and coordination. More specifically, we are developing an extension to the traditional three-layered robot architecture that enables robots to interact directly at each layer — at the behavioral level, the robots create distributed control loops; at the executive level, they synchronize task execution; at the planning level, they use market-based techniques to assign tasks and allocate resources. The market-based planning layer of each robot has two main components: (1) a trader that participates in the market, auctioning and bidding on tasks; (2) a scheduler that determines task feasibility and cost for the trader, and interacts with the executive layer for task execution. This paper focuses on the planning level, detailing the architecture, our current implementation, and planned future extensions. We show how the architecture (in particular, the planning layer) has been applied to a Mars exploration scenario involving the characterization of scientifically "interesting" rocks. We also present preliminary simulation results exploring market and scenario parameters.

Keywords: multi-robot control, market economy, stochastic scheduling, distributed three-layered architecture, Mars exploration

1. Introduction

An architecture for multi-robot control and coordination must be able to accommodate issues of synchronization and cooperation under a wide range of conditions and at various levels of granularity and timescale. An important characteristic for a multi-robot architecture is the flexibility to accommodate different levels of coordination and synchronization. Equally important is the flexibility to take advantage of the resources available to the system, as well as to accommodate the requirements and constraints imposed on it, in order to provide efficient solutions.

In designing a multi-robot architecture, one must inevitably negotiate the tradeoffs between centralized and distributed approaches. A centralized system can make optimal decisions about subtle issues involving many robots and many tasks. In contrast, a highly distributed system can quickly respond to problems involving one (or a few) robots, and is more robust to point failures and the changing dynamics of the system. The flexibility to blend the advantages of both approaches is another desirable trait in an architecture.

We are developing a multi-robot coordination architecture that addresses these issues, providing the flexibility to accommodate: (1) different granularities and timescales of synchronization and coordination; (2) a wide range of system resources, constraints, and requirements while producing reasonable solutions; and (3) the strengths of both distributed and centralized approaches, allowing optimization (when required and possible) while maintaining robustness and the ability to adapt to the (changing) dynamics of the system.

The architecture is an extension of the traditional three-layered approach, which provides event handling at different levels of abstraction through the use of behavioral, executive, and planning layers. Our approach extends the architecture to multiple robots by allowing robots to interact directly at each layer (see Figure 1). This provides several benefits, including: (1) plan construction and sharing using a distributed market-based approach, providing for various degrees of optimization; (2) explicit executive-level, inter-robot synchronization constraints; and (3) distributed behavior-level feedback loops for both loosely- and tightly-coupled coordination.

The focus of this paper is on the market-based planning mechanism for allocating tasks and resources. It is a distributed mechanism that allows for different levels of optimization in the initial allocation-auctions and, if possible, during subsequent refinement-auctions. After first presenting the major components of the architecture, we focus on the details of the market-based planning layer, its two main components (*traders* and *schedulers*) and the interactions between them. We present experimental results from a Mars exploration scenario exploring some of the major parameters of the market mechanism within very controlled round robin auctions.

2. Related Work

Our work blends the advantages of both centralized and distributed approaches to multi-robot systems. Though inherently distributed, the architecture allows the market-based planning layer to utilize different optimization mechanisms which may be centralized to varying degrees. In purely centralized approaches, a centralized plan details the actions for each robot. For example, a planner might treat two 6 DOF arms as a single 12 DOF system for the purpose of generating detailed trajectories (Khatib, 1995). While this approach provides for close coordination, it usually employs centralized monitoring and, thus, suffers from single point failure and lack of local reactivity. At the other end of the spectrum, in the distributed approach (Balch and Arkin, 1994; Mataric, 1992; Parker, 1998), each agent is autonomous, but there is usually no explicit synchronization among the robots.

(Jennings and Kirkwood-Watts, 1998) have developed a distributed executive for multi-robot coordination. The executive, based on a distributed dialect of Scheme, is similar to our executive language in the types of synchronization constructs it supports. As with our work, this enables robots to solve local coordination problems without having to invoke a high-level planner.

Several researchers have investigated economy-based architectures applied to multi-agents systems (Sandholm and Lesser, 1995; Sycara and Zeng, 1996; Wellman and Wurman, 1998), beginning with work on the Contract Net (Smith, 1980). (Stentz and Dias, 1999) proposed a market-based approach that aims to opportunistically introduce pockets of centralized planning into a distributed system. (Thayer et al., 2000), (Gerkey and Matarić, 2001), and (Zlot et al., 2002) have since presented market-based multi-robot coordination results. Our approach differs in that the market-based planning mechanism is part of a larger distributed three-layered architecture. Additionally, in our approach, the market component of each robot is closely associated with a scheduler that schedules tasks, manages resource constraints, and calculates costs, removing this burden from the market.

Our scheduling algorithm falls into the class of *stochastic sampling* algorithms (Langley, 1992; Bresina, 1996; Ruml, 2001; Cicirello and Smith, 2002). These algorithms are memoryless, non-systematic, and make each search decision randomly. The simplest of these algorithms, called *iterative sampling*, makes unbiased decisions at random (Langley, 1992). More sophisticated approaches incorporate heuristics to bias the search. Examples include *Heuristic-Biased Stochastic Sampling* (Bresina, 1996), and our value-biased variant called WHISTLING (Cicirello and Smith, 2002) used in our architecture's planning layer.

Before presenting the details of our market-based approach, we first describe the larger context in which it functions: our distributed three-layered architecture.

3. Architecture Overview

Our multi-robot architecture is based on the layered approach that has been adopted for many single-agent autonomous systems (Bonasso et al., 1997; Muscettola et al., 1998; Simmons et al., 1997). These architectures typically consist of a planning layer that decides how to achieve high-level goals, an executive layer that sequences tasks and monitors task execution, and a behavioral layer that interfaces to the robot's sensors and effectors.

Information and control flows up and down between layers. The planning layer sends plans to the executive, which further decomposes tasks into subtasks and dispatches them based on the temporal constraints imposed by the plan. Dispatching a task often involves enabling or disabling various behaviors. The behaviors interact to control the robot, sending back sensor data and status information. The executive informs the planner when tasks are completed, and may further abstract sensor data for use by the planner. The executive also monitors task execution: in case of failure, it can try to recover or it can terminate the task and request a new plan from the planner.

We extend this architectural concept to multiple robots in a relatively straightforward way. Each robot is composed of a complete three-layered architecture. In addition, each of the three layers can interact directly with the same layer of other robots (Figure 1). Thus, each robot can act autonomously at all times, but can coordinate (at multiple levels) with other agents, when needed. By allowing each layer to interact directly with its peers, we can form distributed feedback loops, operating at different levels of abstraction and at different timescales. In particular, the behavioral layer coordinates behaviors, the executive layer coordinates tasks, and the planning layer coordinates/schedules resources. In this way, problems arising can be dealt with at the appropriate level, without having to involve higher layers. This decreases latency and may increase robustness.

The behavioral layer consists of real-time sensor/effector feedback loops. By connecting the sensor behaviors of one robot to the effector behaviors of another, we can create efficient distributed servo loops (Simmons et al., 2000). Similarly, by connecting effector behaviors to-

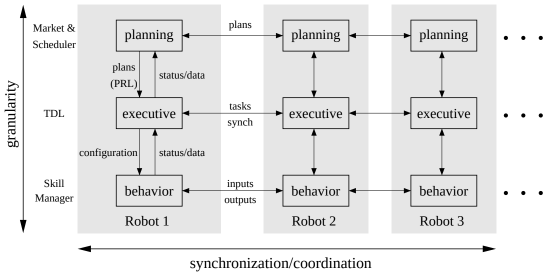

Figure 1. Layered multi-robot architecture

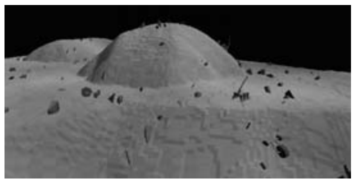

Figure 2. Simulated Mars environment with multiple rovers and rocks.

gether, we can create tightly coordinated controllers. Our implementation extends the Skill Manager of (Bonasso et al., 1997) to provide for both intra- and inter-robot connections. Skills (behaviors) are connected via input/output ports and operate in a data-flow fashion: when a new input value arrives on a port, the skill runs an action code that (optionally) produces outputs.

The executive layer has responsibility for hierarchically decomposing tasks into subtasks, enforcing synchronization constraints between tasks (both those imposed by the planner and those added during task decomposition), monitoring task execution, and recovering from exceptions (Simmons, 1994). Our executive layer is implemented using the Task Description Language (TDL). TDL is an extension of C++ that contains explicit syntax to support hierarchical task decomposition, task synchronization, execution monitoring, and exception handling (Simmons and Apfelbaum, 1998). Recently, we have extended TDL to handle task-level coordination between robots transparently and to enable one robot to spawn or terminate a task on another.

The following section provides more detail about the market and scheduler components of the planning layer.

4. Planning Layer: Market and Scheduler

Our approach to task allocation and planning is based on a market economy. An *economy* is essentially a population of agents coordinating with each other to produce an aggregate set of goods. *Market economies* are generally unencumbered by centralized planning, instead leaving individuals free to exchange goods and services and enter into contracts as they see fit. Conspicuously absent from the market approach is a rigid, top-down hierarchy. Instead, individuals organize themselves in a way that is mutually beneficial. Despite the fact that individuals in the economy act only to advance their own self-interests, the aggregate effect is a highly productive society.

We have developed a market mechanism, based on our work with TraderBots (Stentz and Dias, 1999; Dias and Stentz, 2000; Dias and Stentz, 2001), that is flexible enough to accommodate different auction mechanisms (though in the current implementation, tasks are allocated based on exchanges of single tasks between pairs of robots (Dias and Stentz, 2001)). A robot/agent that needs a task performed announces that it will auction off the task as a seller. Each buyer capable of performing the task for a cost c, bids to do so for $c + \epsilon$. The buyer accepts the lowest bid, as long as it is cheaper than doing the task itself. If a bid is accepted, the seller performs the task and pockets ϵ as profit.

Each robot aims to maximize its individual profit (which often translates to minimizing individual cost where possible); however, since all revenue is derived from satisfying team objectives, the robots self-interest translates to global efficiency. Moreover, the robots can only increase their profit by eliminating unnecessary waste (i.e., excess cost).

The market approach has a number strengths, including: the flexibility of allowing robots to cooperate and compete as necessary to accomplish a task; its amenability to learning new behaviors and strategies during execution of complex global task; and its ability to deal opportunistically with dynamic environments.

In addition to the market component, or trader, of each robot, called the *RoboTrader*, the market of our architecture also contains *OpTraders*, traders that are similar in function except that: they are not associated with a robot, they provide a user interface to the system, and they act on behalf of the operators/users. When a user introduces a task into the system, it is the OpTrader that decomposes the task into its components, if necessary, and initially auctions off the task(s) while trying to minimize cost. In the current implementation, RoboTraders are prevented from participating in more than one external auction at a time. Furthermore, RoboTraders will always set their own auction deadlines to expire after any auction in which they are participating. These constraints ensure that tasks can be costed independently and accurately.

The RoboTrader works closely with the other major component of the planning layer: the scheduler. Following the market economy framework, a scheduler associated with each robot is responsible for maintaining the robot's current agenda of accepted and pending tasks. The scheduler plays a critical role both in the formation of bids and in the interaction between the planning and executive layers. Before a RoboTrader can bid on a new task, it must first ascertain from the scheduler whether the task can in fact be feasibly undertaken (given resource/timing constraints and the other tasks already in the schedule) and, if so, the cost. Once a RoboTrader is awarded a task, it is added to the schedule and now further constrains decisions to bid on other tasks. The scheduler is responsible for sending the task to the executive so that it is executed properly with respect to the other tasks that the robot must accomplish. Executing tasks may also be terminated and sold or rescheduled to reduce costs.

The scheduler is faced with a limited amount of computation/time in which to evaluate the cost for adding tasks to the schedule and the savings for removing them (both of which are needed by the RoboTrader). Rather than attempt to ensure the optimality of the schedules produced, we instead employ an incomplete, non-systematic stochastic search procedure with the goal of quickly finding solutions that are "good enough." Our search algorithm, used by the scheduler, is called *Whistling* (Cicirello and Smith, 2002).

In the current implementation of the architecture, we are not considering temporal constraints between or among tasks. We also assume that a robot can execute only one task, consuming all of the robot's resources. Given these constraints, the scheduling problems faced by the scheduler are essentially traveling salesman problems (TSP). Each task has a location and an estimated duration, and it is the job of the scheduler to find a task sequence that minimizes the sum of the task durations and total travel time.

It should be noted that our goal here is not to develop the next best TSP algorithm. Rather, we have focussed on developing a search procedure that is flexible in design and will easily extend to other, more difficult scheduling problems. In the future, we plan to include hard time constraints and allow multiple overlapping resources and tasks in the scheduling problem. The flexibility and anytime characteristics of Whistling should help facilitate this extension.

The division of the planning layer of each robot into two components (the scheduler and RoboTrader) provides a clean functional separation that also highlights the flexibility of the architecture. The scheduler can incorporate a range of different search heuristics (e.g., as functions of the perceived drivers of task cost), and can be augmented to accommodate different resource constraints. The trader can be augmented with various auction mechanisms, from the distributed mechanism of trades between pairs of robots (currently implemented), to a more centralized combinatorial exchange mechanism.

The next section presents our experimental Mars rock characterization scenario, as well as results testing some of the basic parameters of our market mechanism.

5. Experiments and Results

5.1 Rock Characterization Task

Our Mars exploration scenario is premised on the notion of scientific return, i.e., that a group of robots would be sent to Mars for the (potentially) valuable information they gather and return to Earth. We envision a scenario where a colony of heterogeneous robots is deployed on Mars. Scientists on Earth communicate high-level task descriptions to the colony (e.g., "find and gather data on several carbonate rocks"). We assume that communications limitations (bandwidth, delays, blackouts) necessitate highly autonomous robots, and preclude effective teleoperation of the robots or micro-managing of task execution by the scientists. The robots are therefore responsible for deciding which/how tasks are to be accomplished based on, among other things, the tasks' relative priorities. The goal for the robots is to utilize their time, resources, and capabilities efficiently so as to provide the highest possible scientific return on the tasks they are given.

In terms of the development and testing of our current system, we have focussed on a characterize-region task that will fit within a broader exploration scenario. In this task, a user/scientist specifies a region on the Mars surface, indicating that rocks within that region are to be characterized with an appropriate sensing instrument. The scientist may also specify the locations of rocks, if known. With respect to testing, a 3D graphical simulator developed for the project currently provides the "physical" robots and environment required (Figure 2), though, in the future, we hope to use real robots as well.

5.2 Experimental Results

The goal of the experiments presented here was to examine the effects of some of the major parameters of our market-based planning layer. These experiments represent some of the earliest empirical results of our system, and as such, use a fairly simple experimental scenario and tightly coordinated auction synchronization. All of the experiments used the same initial setup: **6 rovers** start clustered near the center of a rock field; **1 OpTrader** initially allocates **50 rocks** with known locations. The RoboTraders and/or OpTrader participate in a round robin series of auctions, where each trader has the exclusive right to hold an auction during its turn. While not necessarily efficient or realistic, the round robin auction-synchronization mechanism allows: reduction of potentially confounding auction effects; the ability to determine auction quiescence (i.e., no auctions after a one or two rounds) and consequently to delay execution until auction quiescence.

The specific experimental parameters we examined were:

- **Maximum Awards Per Auction (MAPA)**: the maximum number of tasks awarded by the OpTrader in a single auction.
- **No RoboTrader Auctions (NRTA)**: RoboTraders never auctioning their tasks.
- **OpTrader Awards Randomly (OTrand)**: awards made randomly, not based on bids.
- **OpTrader Auctions First (OTfirst)**: the OpTrader auctioning off all of its tasks before the RoboTraders auction any.
- **OpTrader Auctions Normally (OTnorm)**: the OpTrader participating in the round robin auctions normally.
- **OpTrader Auctions After Quiescence (OTafter)**: the OpTrader auctioning only if the previous round had no tasks awarded.

For each of the MAPA values of 1, 3, and 6, we performed experiments with each of the five other parameters, giving 15 experimental combinations.

We ran 5 trials of each combination, with the completion criterion that all 50 rocks be characterized. The experimental data collected for each trial included: the final total time cost of the solution and the number of auctions required (Table 1). Auctions took 8 seconds each, and for each trial were correlated to time-to-completion (data not presented) with $\rho \approx 0.98$. In all experiments, task execution was delayed until auction quiescence.

As a baseline for comparison with Table 1, we ran a number of experiments with both NRTA and OTrand that consistently produced costs of over 1200 across all three MAPA values. Also, experiments to find an optimal solution to our problem using a genetic algorithms (GA) approach produced a best total cost of 777.

5.3 Discussion

There are a number of general trends in the data that are notable in Table 1. One trend is that the quality of the solution degrades (i.e., total cost increases) as the MAPA value increases. The reasons for this is fairly straightforward. In each auction in our system, the OpTrader never awards more than a single task to each RoboTrader bidder, possibly producing inefficiencies. For example, if the OpTrader awards three tasks in a particular auction (MAPA value ≥ 3), they are to three different RoboTraders, when a better allocation might have had two of the tasks

	Mean Total Cost			*Mean Number of Auctions*		
MAPA	1	3	6	1	3	6
NRTA	794 (1.9)	805 (6.0)	967 (11.8)	50.0 (0)	17.0 (0)	9.0 (0)
OTrand	903 (16.7)	939 (36.5)	960 (26.2)	233.8 (26.8)	103.4 (3.9)	69.4 (5.5)
OTfirst	795 (4.6)	807 (6.2)	922 (3.6)	60.8 (1.1)	30.2 (2.7)	23.0 (2.7)
OTnorm	796 (2.8)	804 (7.1)	848 (20.9)	279.6 (3.6)	111.8 (5.3)	67.6 (1.3)
OTafter	798 (4.7)	797 (7.0)	820 (12.9)	287.2 (9.1)	127.2 (5.4)	108.0 (3.1)

Table 1. Mean total cost, and mean number of auctions before market quiescence and execution. Standard deviations in parentheses.

going to the same bidder. Thus, smaller MAPA values tend to lead to better initial allocations, though more OpTrader auctions must be held.

A good solution, given the current limitations of our system, relies heavily on the quality of the initial allocation made by the OpTrader (i.e., with a low MAPA value). Clearly, the random allocation in the OTrand experiments does not qualify. When MAPA=1, the allocations provided by OTfirst, OTnorm, OTafter, are essentially equally good. When the initial OpTrader allocation degrades (at MAPA values of 3 and 6), having the RoboTraders participating with the OpTrader in round robin auctions improves the solution, as is evident from comparing NRTA and OTfirst to OTnorm and OTafter.

A key tradeoff in the data is between the total cost of the solution, the MAPA value, and the number of auctions required for the solution. When MAPA=1, the OpTrader achieves an efficient solution in only 50.0 auctions. Adding many additional RoboTrader auctions does not improve this solution. Similarly, in the MAPA=3 case, the OpTrader achieves a good solution with few (i.e., 17.0) auctions. An improvement requires many additional auctions (127.2 in the OTafter case). When the MAPA value increases to 6, the overall solution quality degrades, but the worth of performing significant numbers of RoboTrader auctions increases.

6. Conclusions and Future Work

With these experiments, we have begun to probe some of the parameters and possibilities of our market-based planning mechanism. As our results demonstrate, both the OpTrader's initial allocation strategy and the use of RoboTrader auctions greatly impact the quality of solutions obtained. It is our hope that these results will begin to provide an understanding of how to use our market-based planning layer effectively under a broad set of domains and conditions.

In the future, we plan to expand the market mechanism of our architecture to allow for opportunistic optimization through the use a combinatorial exchanged mechanism and bids of bundled tasks, as opposed to the current implementation with single-task bids. In addition, we plan to expand the scheduler capabilities to handle absolute timing constraints on tasks, as well as overlapping tasks with multiple, differing resource constraints. We will continue to examine the parameters of our market-based planning mechanism, testing its flexibility, and exploring the fundamental issues and tradeoffs that arise in the system.

Acknowledgments

This work has been supported by several grants, including NASA NCC2-1243, NASA NAG9-1226, DARPA N66001-99-1-892, and DARPA DAAE07-98-C-L032. Thanks go to Jeff Schneider and Drew Bagnell for their comments and advice on the architecture, and costing functions. David Apfelbaum implemented and helped design TDL, and computed the GA baseline. Stuart Anderson implemented the simulator.

References

Balch, Tucker and Arkin, Ronald C. (1994). Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52.

Bonasso, R.P., Kortenkamp, D., Miller, D.P., and Slack, M.G. (1997). Experiences with an architecture for intelligent, reactive agents. *Journal of Artificial Intelligence Research*, 9(1).

Bresina, J. L. (1996). Heuristic-biased stochastic sampling. In *Proc of the 13th National Conference on AI (AAAI-96)*, pages 271–278.

Cicirello, Vincent A. and Smith, Stephen F. (2002). Amplification of search performance through randomization of heuristics. In *Principles and Practice of Constraint Programming – CP 2002: 8th International Conference, Proceedings*, pages 124–138.

Dias, M. Bernardine and Stentz, Anthony (2000). A free market architecture for distributed control of a multirobot system. In *Proc. International Conference on Intelligent Autonomous Systems*, pages 115–122.

Dias, M. Bernardine and Stentz, Anthony (2001). A market approach to multirobot coordination. Technical Report CMU-RI-TR-01-26, Robotics Institute, Carnegie Mellon University.

Gerkey, Brian P. and Matarić, Maja J. (2001). Sold!: Market methods for multi-robot control. In *IEEE Transactions on Robotics and Automation Special Issue on Multi-Robot Systems.*

Jennings, J. and Kirkwood-Watts, C. (1998). Distributed mobile robotics by the method of dynamic teams. In *Proc. Conference on Distributed Autonomous Robot Systems*.

Khatib, O. (1995). Force strategies for cooperative tasks in multiple mobile manipulation systems. In *Proc. International Symposium of Robotics Research*.

Langley, P. (1992). Systematic and nonsystematic search strategies. In *AI Planning Systems: Proc of the 1st International Conference (AIPS-92)*.

Mataric, M. (1992). Distributed approaches to behavior control. In *Proc. SPIE Sensor Fusion V*, pages 373–382.

Muscettola, N., Nayak, P. P., Pell, B., and Williams, B. (1998). Remote agent: To boldly go where no ai system has gone before. *Artificial Intelligence*, 103(1–2):5–48.

Parker, L. (1998). Alliance: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

Ruml, W. (2001). Incomplete tree search using adaptive probing. In *Proc of the 17th International Joint Conference on AI (IJCAI-01)*, pages 235–241.

Sandholm, Toumas and Lesser, Victor (1995). Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *Proc. International Conference on Multiagent Systems*, pages 328–335.

Simmons, R., Goodwin, R., Haigh, K., Koenig, S., and O'Sullivan, J. (1997). A layered architecture for office delivery robots. In *Proc. 1st International Conference on Autonomous Agents*.

Simmons, Reid (1994). Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43.

Simmons, Reid and Apfelbaum, David (1998). A task description language for robot control. In *Proc. International Conference on Intelligent Robots and Systems*.

Simmons, Reid, Singh, Sanjiv, Hershberger, David, Ramos, Josue, and Smith, Trey (2000). First results in the coordination of heterogeneous robots for large-scale assembly. In *Proc. International Symposium on Experimental Robotics*.

Smith, R. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113.

Stentz, Anthony and Dias, M. Bernardine (1999). A free market architecture for coordinating multiple robots. Technical Report CMU-RI-TR-99-42, Robotics Institute, Carnegie Mellon University.

Sycara, K. and Zeng, D. (1996). Coordination of multiple intelligent software agents. *International Journal of Cooperative Information Systems*, 5(2–3).

Thayer, Scott, Digney, Bruce, Dias, M. Bernardine, Stentz, Anthony, Nabbe, Bart, and Hebert, Martial (2000). Distributed robotic mapping of extreme environments. In *Proc of SPIE: Mobile Robots XV and Telemanipulator and Telepresence Technologies VII*.

Wellman, Michael and Wurman, Peter (1998). Market-aware agents for a multiagent world. *Robotics and Autonomous Systems*, pages 115–125.

Zlot, Robert, Stentz, Anthony, Dias, M. Bernardine, and Thayer, Scott (2002). Multi-robot exploration controlled by a market economy. In *Proc. International Conference on Robotics and Automation*.